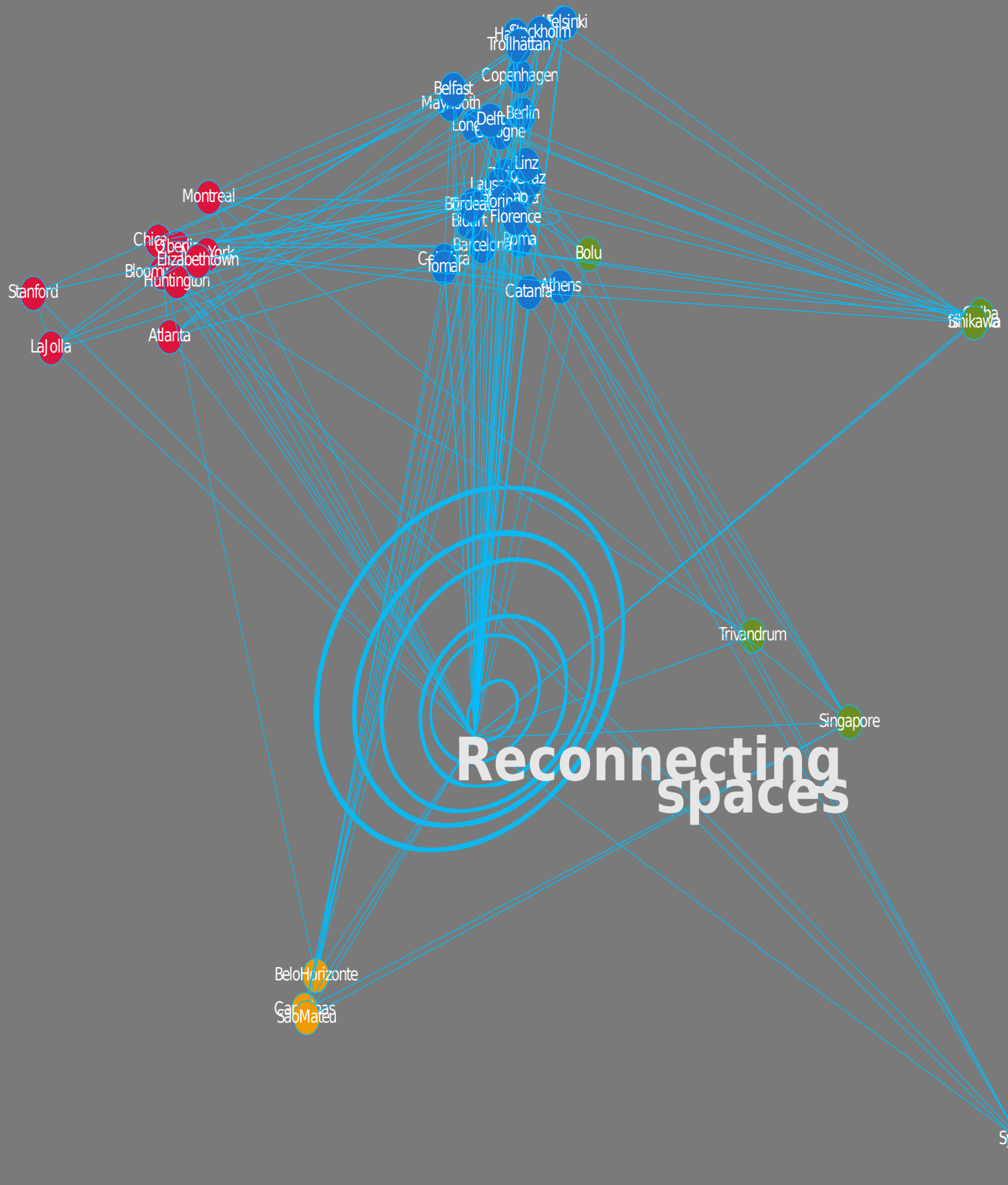
18th Sound and Music Computing Conference

June 29th - July 1st 2021

Proceedings



SMC 2021

Proceedings of the 18th Sound and Music Computing Conference

June 29th – July 1st 2021

Davide Andrea Mauro, Simone Spagnol and Andrea Valle, eds.

18th Sound and Music Computing Conference
June 29th – July 1st 2021

Organization:
SMC – Sound and Music Computing Network - `http://www.smcnetwork.org`

With the support of:


UNIVERSITÀ DEGLI STUDI DI TORINO


CONSERVATORIO STATALE DI MUSICA GIUSEPPE VERDI TORINO


POLITECNICO DI TORINO

sponsored by

Proceedings of the 18th Sound and Music Computing Conference
Davide Andrea Mauro, Simone Spagnol and Andrea Valle, editors
ISBN: 978-88-945415-4-0
ISSN: 2518-3672
website: http://www.smc2021conference.org/

BibTeX:

```
@proceedings{18SMCConf,
Editor = {Davide Andrea Mauro, Simone Spagnol and Andrea Valle},
Organization = {Sound and Music Computing Network},
Publisher = {Axea sas/SMC Network},
Title = {Proceedings of the 18th Sound and Music Computing Conference},
Year = {2021}}
```


Typeset with ConTEXt by Andrea Valle

General Chairs
Simone Spagnol – TU Delft, The Netherlands
Davide Andrea Mauro – Marshall University, Huntington, USA
Andrea Valle – University of Torino, Italy

Scientific Committee
Federico Avanzini – University of Milano, Italy
Stefan Bilbao – University of Edinburgh, UK
Jean Bresson – Ableton, Berlin, Germany
Emma Frid – Royal Institute of Technology, Stockholm, Sweden
Emilia Gómez – Universitat Pompeu Fabra, Spain
Vincenzo Lombardo – University of Torino, Italy
Marcella Mandanici – Music Conservatory "L. Marenzio", Italy
Georgios Marentakis – Østfold University College, Norway
Andrew McPherson – Queen Mary University London, UK
Romain Michon – GRAME-CNCM, France & CCRMA, Stanford University, USA
Juhan Nam – Korea Advanced Institute of Science and Technology, South Korea
Sandra Pauletto – Royal Institute of Technology, Stockholm, Sweden
Gary Scavone – McGill University, Canada
Sebastian Schlecht – Aalto University, Finland
Stefania Serafin – Aalborg University Copenhagen, Denmark
Louena Shtrepi – Polytechnic University of Torino, Italy
Isabelle Viaud-Delmon – IRCAM/CNRS, France

Panel of Scientific Reviewers
Andrea Agostini, Jack Armitage, Adrián Barahona-Ríos, Adriano Baratè, Jonathan Bell, Edgar Berdahl, Laura Bishop, Charles Brazier, Gary Bromham, Grigore Burloiu, Eoin Callery, Emilios Cambouropoulos, Elliot Canfield-Dafilou, F. Amílcar Cardoso, Thibaut Carpentier, Alexander Carôt, Chris Chafe, Vasileios Chatzioannou, Stefano D'Angelo, Matthew Davies, Yuri De Pra, Stefano Delle Monache, Michele Ducceschi, Cagri Erdem, Fabian Esqueda, Georg Essl, Laurent Feisthauer, Aníbal Ferreira, Leonardo Fierro, Federico Fontana, Francesco Foscarin, Martin Gasser, Daniele Ghisi, Sergio Giraldo, Kjetil Falkenberg Hansen, Andrew Horner, Daniel Hug, Alexander Refsum Jensenius, Pierre Jouvelot, Hanna Järveläinen, Haruhiro Katayose, Boris Kuznetsov, Olivier Lartillot, James Leonard, Hans Lindetorp, Luca Andrea Ludovico, Raul Masu, Benjamin Matuszewski, Daniel Mayer, Riccardo Miccini, Nicolas Misdariis, Dave Moffat, Fabio Morreale, Niels Christian Nilsson, Hiroki Nishino, Stavros Ntalampiras, Daniel Overholt, Elif Özcan, Razvan Paisa, Rui Pedro Paiva, Claudio Panariello, Stefano Papetti, Julian Parker, Johan Pauwels, Pietro Polotti, Pedro J. Ponce De León, Karolina Prawda, Giorgio Presti, Niccolò Pretto, Marcelo Queiroz, Davide Rocchesso, Gerard Roma, David Roze, Charalampos Saitis, Mark Sandler, Giovanni Santini, Diemo Schwarz, Rod Selfridge, Federico Simonetta, Julius Smith, Bob Sturm, Burak Tamer, Erica Tavazzi, Sten Ternström, Satoshi Tojo, Luca Turchet, Maarten Van Walstijn, Satvik Venkatesh, Gerhard Widmer, Silvin Willemsen, Minz Won, Asterios Zacharakis

Table of Contents

Session 3 - Sound In Space I: Modeling And Rendering

Session 4 - Physical Modeling

Session 5 - Systems For Music Generation And Composition

Session 6 - Sound Synthesis

Session 7 - Music Classification, Recommendation, And Preservation

Preface

The Sound and Music Computing Conference reaches its 18^{th} edition!

This 18th Sound and Music Computing Conference (SMC 2021) is organized under the auspices of the Sound and Music Computing Steering Committee.

SMC 2021 is an interdisciplinary forum to share music, thoughts, needs and discoveries in this remarkable research topic that brings together art, technology, and human perception. SMC 2021 Topics of Interest include a wide selection of topics related to acoustics, psychoacoustics, technologies for audio and music, audio analysis and synthesis, spatial sound, sonic interaction design, music analysis, performance modelling, and many more.

In the original call for papers and works, SMC 2021 welcomed two types of contributions:

- Scientific contributions examining all the core topics of the Sound and Music Computing field; these contributions, which have been fully peer-reviewed, are presented as oral talks.
- Music contributions that make use of the possibilities that technology offers nowadays to create music in a broad sense.

SMC 2021 received 123 full submissions: 79 scientific contributions, and 44 musical contributions.

Out of them, SMC 2021 features 50 oral presentations and 18 musical pieces.

SMC 2021 had the help of 92 scientific reviewers that performed 342 reviews to examine all the submissions in order to compile the final Scientific Program. Based on recommendations from the Scientific Committee, the Scientific Chairs have made the final decisions and organized the presentation of the different contributions in twelve different Sessions.

In addition, since for two years in a row we missed the opportunity to visit the locations where our community is working, we later welcomed a new type of contribution: Lab Presentations. Nine institutions let us – virtually – in, allowing us to visit their spaces and learn more about their activities and research.

SMC 2021 features four keynote speakers (Chris Chafe, Scot Gresham-Lancaster, Emilia Gómez, Michele Ducceschi), all relevant members of the community in terms of scientific and artistic research. Their contributions show the diversity of the community itself.

Finally, the music program is rich. It features a special event dedicated to David Cope, and two open calls: a networked concert, and the screening of silent film sonorizations. The latter is a project in collaboration with the National Museum of Cinema that we had started last year and that we complete this year.

The conference takes place in a virtual format between June the 29th and July the 1st 2021.

We really thought we could leave COVID-19 behind us in 2020 but the persisting situation has forced us to rethink our 2021 edition. We strongly supported the decision of keeping the yearly schedule of the conference.

"Did we learn anything from COVID-19?". This question is still very much on the table.

Surely, we have become acquainted -even obsessed- with online resources. But we have also gained awareness of the inescapable need to meet personally, physically. So, the theme of SMC 2021 is "Reconnecting spaces". Even if forced again, hopefully for the last time, to meet online, our effort and hope is to keep our community (a worldwide one) tightly integrated, and also to expand it. Having organized the conference online since the first call, we have received contributions from researchers that probably would never have come to the physical edition.

For the second time this year, we are making a choice which is rooted in science and in politics. Is it useful to keep access to an online conference reserved to a closed group of registered participants? We do not think so. Again, this 2021 edition is open to anyone with an internet connection.

Huntington/Delft/Torino, June 2021
Davide Andrea Mauro, Simone Spagnol and Andrea Valle
SMC 2021 General Chairs

Keynotes

Keynote 1

Chris Chafe

Unlocking Musical Performances During the Lockdowns

The presentation will feature performances from the year of COVID-19 quarantines largely focused on how traditional ensembles could be reconstituted online from home. New technical work and new discoveries about the capabilities of today's network and computing infrastructure have happened with contributions from volunteer code contributors companies, a new foundation, and several computer music research centers. Above all improvements have been driven by the musicians who have taken part. The pandemic has ushered in a new phase of development driven by musicians seeking solutions, particularly ease of use and the ability to scale across worldwide cloud infrastructure. With orchestral-sized ensembles urgently in need of ways to rehearse on the network and most participants running their systems over commodity connections, this "new reality" runs counter to what's required for ultra-low-latency rhythmic synchronization. JackTrip which has generally been run as a native software application is now complemented by dedicated solutions including numerous Raspberry Pi-based systems, standalone physical web devices, and browser-based WebRTC and Pure Data versions. I conclude with some thoughts about how in our physical realms we're creatures who listen and function with inherent delays and Internet Acoustics is a new realm into which we're expanding. Pre-COVID, that was more on the level of a thought experiment and now we're accelerating towards it.

About the speaker

Chris Chafe is a composer, improvisor, and cellist, developing much of his music alongside computer-based research. He is Director of Stanford University's Center for Computer Research in Music and Acoustics (CCRMA). In 2019, he was International Visiting Research Scholar at the Peter Wall Institute for Advanced Studies The University of British Columbia, Visiting Professor at the Politecnico di Torino, and Edgard-Varèse Guest Professor at the Technical University of Berlin. At IRCAM (Paris) and The Banff Centre (Alberta), he has pursued methods for digital synthesis, music performance and real-time internet collaboration. During the pandemic he's released an album, "Time Crystal" on Ravello Records, performed over 60 concerts online and been a contributor to a large volunteer effort for improvements to network music performance. At CCRMA he is involved in research into wavefield synthesis for physical models and learning from his co-workers about deep machine learning networks for music prediction and how quantum computing technologies can be introduced into music making.

`http://chrischafe.net/`

Keynote 2

Scot Gresham-Lancaster

Computer Network Music - An Examination of the Roots of a New Genre of Computer Music

A new genre of music practice where interactions of networks of personal computers generate note and sound choices is described. It is the speakers feeling that this approach grew directly out of cultural sense of a collective technological utopia. This approach was realized by the availability of personal computer technology and networking. Initially practiced by a community of electroacoustic composer/performers from the San Francisco Bay Area circa 1978, it spread to become encompassed in practices of many laptop composers in a variety of ways. There is an important distinction to between work made between heterogeneous collectives starting with the League of Automatic Music Composers and homogeneous "Laptop Orchestras".

About the speaker

Scot Gresham-Lancaster is a composer, performer, instrument builder, and educator. He is a Research Scientist with the startup StrangeData LLC and Visiting Researcher at CNMAT UC Berkeley. The focus of his research is in the sonification of data sets in tight relationships with visualizations, (multimodal representations). As a member of the HUB, he is an early pioneer of networked computer music and has developed many "cellphone operas". He has created a series of co-located international Internet performances and worked developing audio for several games and interactive products. He is an expert in educational technology.

`http://scot.greshamlancaster.com/`

Keynote 3

Emilia Gómez

TROMPA: Towards Richer Online Music Public-domain Archives

In this talk, I will present the main approach and outcomes of the TROMPA Horizon 2020 European project, which I have coordinated in the last years with researchers on the use of machine and human intelligence for the enrichment of classical music archives. Classical music, although a historical genre, it is continually (re)interpreted and revitalised through musical performance. TROMPA intends to enrich and democratise publicly available classical music archives through a user-centred co-creation setup. For analysing and linking music data at scale, the project employs and improves state-of-the-art technology. Music-loving citizens then cooperate with the technology, giving feedback on algorithmic results, and annotating the data according to their personal expertise. Following an open innovation philosophy, all knowledge derived is released to the community in reusable ways. This enables many uses in applications which directly benefit crowd contributors and further audiences. TROMPA demonstrates this for music scholars, orchestras, piano players, choir singers, and music enthusiasts.

About the speaker

Emilia Gómez is Lead Scientist of the HUMAINT project that studies the impact of Artificial Intelligence on human behaviour, carried out at the Joint Research Centre, European Commission. She is also a Guest Professor at the Department of Information and Communication Technologies, Universitat Pompeu Fabra in Barcelona, where she leads the MIR (Music Information Research) lab of the Music Technology Group and coordinates the TROMPA (Towards Richer Online Music Public-domain Archives) H2020 project.
Emilia Gómez's work has been involved in the Sound and Music Computing Network for many years, contributing in several roles such as author, reviewer and board member. She has also been serving the ISMIR community, being the first woman president of the International Society for Music Information Retrieval. She is particularly interested in improving gender and cultural diversity of our research field.

`http://www.emiliagomez.com/`

Keynote 4

Michele Ducceschi

Real-time, Large Scale Physical Modelling Sound Synthesis

Physical modelling sound synthesis has long roots. In fact, the first ever example of a partial differential equation dealt with the musical problem of the vibrating string, which puzzled the minds of the most renowned scientists of the mid-1800s. The ideas that came into existence during this "vibrating string controversy" established the foundation of early physical modelling synthesis techniques, from digital waveguides to modal methods. Today, mainstream numerical methods can be employed to solve complex mathematical equations using just a fraction of the available CPU. But things are not straightforward, and considerable effort is spent in the design of suitable integration algorithms. In this talk, a coarse review of the leading ideas in physical modelling sound synthesis will be given. In the second part, illustrative examples of typical objects (oscillators, strings and plates) will be shown. Finally, demos of advanced physical models (a plate reverb, and a spring-bar network called Derailer) will be played. The demos are freely available for download at `www.physicalaudio.co.uk`.

About the speaker

Michele Ducceschi currently serves as Principal Investigator for the European Research Council (ERC) Starting Grant NEMUS. This is a 5-year project aiming at synthesising the sound of historical musical instruments, that are currently out of playing condition. Previously, he was a Leverhulme Early Career Fellow (2017) at the Acoustics and Audio Group at the University of Edinburgh, Scotland. He was also a Royal Society Newton International Fellow (2015) and part of the NESS project. Ducceschi's research deals primarily with the sound synthesis of acoustic instruments by physical modelling. He is particularly interested in the efficient simulation of nonlinear systems, either lumped or distributed, and he is also interested mechanical reverberation.

`http://mdphys.org`

Lab presentations

University of Iceland - Acoustic and Tactile Engineering (ACUTE) Lab

We will present a research and development project which two research groups at the University of Iceland that are working together on. The project is on developing an acousto-vibrotactile solution that with the purpose of increasing music enjoyment of cochlear implant recipients. The two research groups are Acoustic and tactile engineering (ACUTE) located in the faculty of Industrial Engineering, mechanical Engineering and Computer Science and the Vision lab located in the faculty of Psychology.
We will also present other ongoing projects which ACUTE lab is working on, e.g. Virtual acoustics and relating 3D scans of pinnas to measured HRFTs.
Acoustic related research at the University of Iceland is relatively new and for this reason we would really appreciate the opportunity to present our work to SMC2021 attendees.

Aalto University - Acoustics Lab

Tapio Lokki, Nils Meyer-Kahlen, Sebastian Schlecht, and Vesa Välimäki
Aalto University, Acoustics Lab, Department of Signal Processing and Acoustics, Espoo, Finland

The professor's and researchers working at the Aalto Acoustics Lab have prepared video presentations and demos in the special facilities of the laboratory.
The laboratories were renovated in two years ago, so they are now in great shape. Most of the video material is recorded binaurally so that it can be enjoyed well with headphones.
The spaces include the Large Anechoic Chamber named after its designer Lampio, the Variable Acoustics Room Arni (named after Paavo Arni, a Finnish pioneer of variable acoustic design), the Listening Room Otala (named after Prof. Matti Otala, who discovered the transient intermodulation distortion), sound-proof listening booths, and a Multichannel Anechoic Room called Wilska (named after the Finnish developer of an early dummy head in the 1930s).
The Large Anechoic Room is the most silent place in Finland, with a background noise level of -2 dB. Our anechoic demonstrations show the directivity of the human mouth, distance decay, and a balloon pop completely free of reverberation.
In the Variable Acoustic Room Arni, the listeners can appreciate the dramatic changes in the room reverberation caused by opening and closing all or some of the panels in all walls of the room.
The Listening Room demo will reveal what can be achieved with a multichannel audio system containing a dozen large hi-fi loudspeakers. The Multichannel Anechoic Room Wilska has over 40 small loudspeakers installed on a spherical grid.
In the demo, the audience can hear how the same excerpt of orchestral music can be played in different concert halls, such as the Berlin Philharmonie and the Berlin Konzerthaus, using a virtual acoustic system developed at Aalto.
Furthermore, the video presentation will give information about the BSc, MSc, and doctoral education provided in the field of acoustics and audio technology at Aalto University.
All our teaching in MSc and doctoral levels is given in English.

`https://www.aalto.fi/en/aalto-acoustics-lab`

University of Milano - Laboratorio di Informatica Musicale (LIM)

LIM - standing for Laboratorio di Informatica Musicale (Music Informatics Laboratory) - is one of the main labs of the Department of Computer Science, Università degli Studi di Milano. It includes 4 rooms equipped with music, multimedia and computer devices. Established in 1985, under the direction of its founder Goffredo Haus LIM hosted composers and renowned experts such as Angelo Paccagnini, Antonio Josè Rodriguez Selles, Franco Donatoni and Dante Tanzi.
Research Topics:

- Sound & Music Computing
- Multilayer navigation of music contents
- Music Information Retrieval
- Sound synthesis and rendering in interactive contexts
- Technologies for music teaching and learning
- Cultural heritage preservation and exploitation

Standards for Music Representation
The LIM is actively involved in the creation of international standards for the representation of music information. In 2008 it played a key role in the standardization of the IEEE 1599 format, and currently it is leading the revision of the standard. Additionally, the LIM contributes to the revisions of the MIDI standard and to the W3C group on new formats for music notation.

Projects and Collaborations
Along more than 30 years of activities, the LIM has taken part in national and international scientific projects and has established collaboration with several institutions, including: Teatro alla Scala, Bolshoi Theater, RAI Radiotelevisione Italiana, RSI Radiotelevisione Svizzera, Microsoft, Verdi Orchestra of Milan, IEEE Computer Society, Ricordi Historical Archive, Italian Ministry of Cultural Heritage. Among the most notable projects, relevant examples include the digitization and exploitation of the Teatro alla Scala archives, the realization of the sound section of the National Science & Technology Museum in Milan, the release of multimedia products for Pearson, the project Bach Digital with the Leipzig Bach Archiv, the realization of the music collection of the European Library of Information and Culture (BEIC), the research on 3D audio for the NASA Charles Ames Research Center.

Teaching
Within the degrees offered by the Computer Science Department, the LIM is the reference research structure for education in topics related to sound and music computing. The study path is structured as follows:

- Bachelor degree in Music Informatics;
- Master degree in Computer Science, with a major in Music Informatics;
- PhD in Computer Science, with research topics in Sound and Music Computing.

Within the bachelor degree, the LIM staff is in charge of courses in Acoustics, Databases, Economy of Musical Heritage, Informatics for Music, Informatics for Sound, Legal Aspects of Music Information, Methods and Technologies for Music Publishing, Models of Music Perception, Music Semiotics, Digital Technologies for Music Information Restoration. Within the master degree, the LIM staff is in charge of courses in Audio Pattern Recognition, MIDI Programming, Music Programming, Organization and Digitization of Multimedia Archives, Sound in Interaction, Timbre Programming.

Equipment
LIM assets include a research lab (approx. 100 m^2), a silent booth and a large acoustically insulated room with reconfigurable internal acoustics. The lab is equipped with dedicated instrumentation, including devices for the digitization and restoration of analog audio materials, MIDI chains, and electronic musical instruments.

`https://www.lim.di.unimi.it/`

University of Trento - Creative, Intelligent and Multisensory Interactions Laboratory (CIMIL)

The Creative, Intelligent and Multisensory Interactions Laboratory (CIMIL) is one of the research teams within the Department of Information Engineering and Computer Science of University of Trento.
CIMIL is an interdisciplinary research lab committed to research on new forms of interaction and experience, based on emerging technologies. The aim is to be at the forefront of innovation on various sectors, including the industrial, artistic and cultural ones. Our vision is to design and create innovative technologies to solve real-world problems. We believe that it is of crucial importance to understand the needs of people in order to design technologies effectively capable of delivering optimal user experiences and addressing community-driven problems.
To this end, CIMIL promotes an interdisciplinary research culture that brings together diverse areas of interest and inquiry, including technology (engineering, computer science), humanities (social sciences, ethics, experimental psychology), and art (music, interactive installations).
Our work falls into four broad categories:

- *Education*: We teach students to design, develop and evaluate proof of concepts prototypes of interactive systems that have intelligent and multisensory features.
- *Basic research*: We aim to devise novel hardware and software multisensory technology and improve existing ones, as well as understand perceptions, behaviors and emotions of their users.
- *Applied research*: We aim to support and empower specific user groups (both able-bodied and disabled, such as the visually-impaired) by means of creative, intelligent, and multisensory technology.
- *Art and culture*: We aim to explore new forms of artistic expression and production (including performance, pedagogy, composition), as well as promote and preserve cultural heritage.

`https://www.cimil.disi.unitn.it/`

Aalborg University Copenhagen - Multisensory Experience Lab (ME-Lab)

The Multisensory Experience Lab (ME-Lab) is based at Aalborg University Copenhagen, Denmark. Our work is centered on the use of multisensory technologies (e.g., virtual and augmented reality) and falls into three broad categories:

- *Basic research*: We aim to improve multisensory technology and understand its users (e.g. perception, cognition, and affect).
- *Applied research*: We aim to assist and empower specific user groups by means of multisensory technology.
- *Art and culture*: We aim to explore new forms of artistic expression and preserve cultural heritage using multisensory technology.

We are particularly interested in researching topics related to sonic interaction design for multimodal environments, simulating walking experiences, sound rendering and spatialization, haptic interfaces, cinematic VR and evaluation of user experience in multimodal environments.

This video previews a selection of our previous work related to (1) novel music instruments; (2) physical and virtual recreation of historical music instruments; (3) sound and music computing applied to therapy, education, and training; and (4) simulated spaces and real exhibits.

`https://melcph.create.aau.dk`.

University of Padova - Centro di Sonologia Computazionale (CSC)

Sergio Canazza, Giovanni De Poli, Niccolò Pretto, Antonio Rodà, Alvise Vidolin
Centro di Sonologia Computazionale, University of Padova

Research in computer music at the University of Padua, Italy, began in the early 1970s and was formalized in 1979 by establishing the Centro di Sonologia Computazionale (CSC).
During the years, CSC research moved in several directions, pushed by the advancement of technology and knowledge and by researchers' curiosity. In the virtual tour the recent research topics will be shown in the area of musical cultural heritage, multimodal interaction and music research-production.
An important theme is the preservation of musical cultural heritage, in particular for art forms in which technology had played an important role, such as electronic and computer music (in which composers worked directly with magnetic tape) or interactive multimedia installations. It is motivated by the awareness of technological obsolescence and the historical importance of the music works realized at CSC. The main results on preservation and enhancement of musical cultural heritage will be presented, such as philologically informed conservation of audio documents, access tools which recreate the experience of the original analogue devices, installations to virtually play historical and modern musical instruments, preservation methodologies of interactive multimedia installations.
The multimodal interaction research opened up important new societal fields of research, such as inclusive systems dedicated to learning for people with special needs, using modeling for tracking of human motion and nonverbal 3-D sounds as a preferred communication channel. In our opinion, inclusive learning for participants with disabilities is one of the most important and urgent aims in the new millennium.
Since the 2010s, CSC has been developing interactive applications based on large-scale responsive environments and user-friendly involvement with expressive behaviour (for teaching music or for tuition of the visually impaired), emphasizing the added pedagogical value of fun and competition. Regarding music research at CSC, scientists, researchers, and technicians continue to collaborate with artists using the new art-science-interaction laboratory and CSC know-how as a support for the innovation of expressive forms in music, music theater, and interactive multimedia arts. In addition, CSC aims to promote and encourage the production of works that use computer systems to control and to create music, especially projects that use technologies developed in its laboratories.
It has also developed new interfaces to play instruments, necessary to control musical timbre and the virtual space, stimulating the interest of many composers who

believe that the traditional keyboard is not suited to simultaneously control multiple parameters, synthesis algorithms, and sound spatialization.
CSC is striving to be a leader in the ``Society 5.0'' revolution, taking advantage of the fact that music is a trans-cultural language. Currently, the CSC visions are to facilitate (1) the inclusion of people with special needs (e.g., multimodal interaction for learning and well-being, acoustic analysis for safety and security in the workplace) and (2) dialogue among different cultures and populations (production of new cultural events, computing and cultural heritage, preservation and enhancement of audio documents, and computational creativity).

`http://csc.dei.unipd.it/`

KTH Royal Institute of Technology - Sound and Music Computing (SMC) Group

Roberto Bresin, Kjetil Falkenberg, André Holzapfel, Sandra Pauletto
KTH Royal Institute of Technology, Stockholm, Sweden

The SMC Sound and Music Computing group[1] at KTH Royal Institute of Technology in Stockholm is a research group of about 20 people within the Division of Media Technology and Interaction Design, EECS School of Electrical Engineering and Computer Science. The long-term vision of the team is ``to understand human communication and interaction by sound and music so as to make them a natural part of everyday technology''. We work towards our vision through engagement in both educational and research activities.
We give a number of SMC-related courses at all three levels of education: Bachelor, Master and PhD.
We actively involve students in ongoing research both as part of and beyond course activities[2]. Among these activities are pilot experiments, prototype development, public exhibitions, performing, composing, data collection, analysis tasks, and not least bachelor and master thesis projects that lead to academic publications. The major benefit for research as we see it is the increased diversity of the research outcomes.
Our research covers a diversity of areas including music expression modelling, affective computing, sensor-based movement analysis, sonification and data representation, sonic interaction design, non-verbal communication in human robot interaction, methods for media production, sound-based methods for rehabilitation/training, music information retrieval, and computational ethnomusicology.
Our research has been funded through grants from a large set of funding bodies including the European Commission, VR Swedish Research Council, Vinnova, NordForsk, MMW, WASP-HS, Formas, The Swedish Retail and Wholesale Council, Hakon Swenson Stiftelsen, PTS Swedish Post and Telecom Authority, JSPS/Kakenhi, Swedish Energy Agency.
Our research group is part of the Nordic SMC Hub network[3] funded by Nord-

1 Sound and Music Computing group at KTH: https://www.kth.se/hct/mid/research/smc

2 Hansen, Bresin, Holzapfel, Pauletto, Gulz, Lindetorp, Misgeld & Sköld. (2019). Student involvement in sound and music computing research: Current practices at KTH and KMH. Combined Proceedings of the Nordic Sound and Music Computing Conference 2019 and the Interactive Sonification Workshop 2019, 36–42. https://doi.org/10.5281/zenodo.3755825

3 NordicSMC hub https://nordicsmc.create.aau.dk/

Forsk. During the years we have been organizing international conferences (such as the SMC conference) and creating new ones, both national and international. We have a tradition of active participation to several conferences (SMC, NIME, ICAD, ISMIR, ISon, TENOR, Audio Mostly, WAC, INTED, CHI, HRI, TEI, ACII) to which not only senior researchers and PhD candidates present their works but also undergraduate students. We continuously publish in both peer-reviewed international journals and conferences: since January 2019 we have published about 14 journal papers, 59 conference papers, and 5 book chapters.

University of Oslo - RITMO Centre for Interdisciplinary Studies in Rhythm, Time and Motion

Alexander Refsum Jensenius and Stefano Fasciani
University of Oslo

We propose a virtual tour of some music technology labs at the University of Oslo, Norway, including:

- *The Portal*: This is a room set up for researching network music with several low-latency audiovisual streaming systems and exploring the integration with other technologies such as spatial audio, motion capture, and virtual reality. The lab is central to the activities of the Music, Communication and Technology master's programme.
- *The fourMs Lab*: a lab used for studies of music-related body motion. It contains several high-end motion capture and physiological sensing systems. It also has a multichannel audio setup. The lab is used by many researchers at RITMO Centre for Interdisciplinary Studies in Rhythm, Time, and Motion.
- *Musical Robotics Lab*: a lab used for developing new musical robots, including prototyping.
- Additional labs and studios include electronics labs, video and sound studios.

In the video, we will also give an overview of ongoing sound and music computing research activities. These include network music, music information retrieval, new interfaces for musical expression, musical robotics, and embodied music cognition.

`https://www.uio.no/ritmo/english/`

Universitat Pompeu Fabra - Music Technology Group (MTG)

The Music Technology Group (MTG) of the Universitat Pompeu Fabra in Barcelona, part of its Department of Information and Communication Technologies, carries out research on topics such as audio signal processing, music information retrieval, musical interfaces, and computational musicology.
The MTG wants to contribute to the improvement of the information and communication technologies related to sound and music, carrying out competitive research at the international level and at the same time transferring its results to society.
To that goal, and guided by our values, the MTG aims at finding a balance between basic and applied research while promoting interdisciplinary approaches that incorporate knowledge and methodologies from both scientific/technological and humanistic/artistic disciplines.
In our presentation we highlight the main research challenges we are currently working on and we emphasize how we promote social and industrial impact. The senior researchers and staff of the MTG present the various topics and we also show the different spaces of the university that we use.

`https://www.upf.edu/web/mtg`

Contributions

EQUALIZATION OF WOOD-PANEL LOUDSPEAKERS

Juho LISKI (juho.liski@aalto.fi)[1], **Jussi RÄMÖ**[1], **Vesa VÄLIMÄKI**[1], and **Otso LÄHDEOJA**[2]

[1]*Acoustics Lab, Dept. of Signal Processing and Acoustics*, **Aalto University**, Espoo, Finland
[2]*Sibelius Academy*, **University of the Arts Helsinki**, Helsinki, Finland

ABSTRACT

This paper studies the acoustic properties of a tree orchestra consisting of four wood-panel loudspeakers and proposes an equalizer (EQ) design for each loudspeaker. Two design strategies for graphic equalization on Bark bands are considered: a single- and a multi-point approach. Asymmetries in the wood-panel speakers cause their magnitude responses to vary so much in different directions that the multi-point averaged EQ gets smoothed and does not have much effect. A single-point EQ, designed based on the frontal response, changes the magnitude response more and improves the overall shape of the response substantially in front of the panels. The magnitude responses at other measurement points are also improved. The EQ does not attenuate the ringing of the wood-panel modes much, thus retaining their resonant quality. The orchestra of equalized wood-panel speakers is used in a science center to showcase acoustic properties of wood.

1. INTRODUCTION

Contrary to the neutral sound of moving-coil cone loudspeakers, a more natural sound has been pursued with unusual types of sound reproduction methods. An example is the panel loudspeaker comprising a wooden board and actuators, which is studied in this work. This can basically be seen as the elemental form of a distributed-mode loudspeaker [1]. Equalization of such speakers is studied in [2]. Other constructions involving wood-induced sound radiation are, for example, wooden boxes used as loudspeakers, and actuators attached to existing wooden surfaces. All these follow the recent trend of hidden or invisible sound [3–5]. Other materials such as glass have also been tested in speakers [6–9].

The sound generation method studied in this paper is called structure-borne sound, which is typically characterized by resonances, an unusual spatial image caused by a large radiating surface, and sound localization behind the radiating surface [10]. Furthermore, when multiple actuators are attached to a panel, a complex superposition of excited modes is obtained [11]. The resulting frequency response thus contains heavy spectral coloration, and depending on the application, equalization may be beneficial.

Sowden and Ampel [6] reported on the development of professional/commercial planar loudspeakers, where they experimented with various types of radiating surfaces. They found that large surfaces typically attenuated the high-frequency response, whereas a light, radiating panel led to better efficiency. The radiating surface acts as a dome radiator, with a frequency-dependent size, i.e., the lower the frequency, the larger the size of the radiating surface. Furthermore, the directivity varied less than with a conventional loudspeaker, but less predictably.

Berndtsson presented measurements on "acoustic walls" [12] and performed a perceptual study with such systems [13]. The acoustic walls consist of pairs of boxes, where the soundboard is made of a specially-treated spruce [12]. A loudspeaker coil driven by a large magnet is attached to the soundboard, and it is fed by a compressed and equalized microphone signal. Since their aim was to improve room acoustics by adding more reverberation, the resonance and sound radiation properties of the acoustic walls were analyzed. The measurements showed that the system deviated from an ideal one, and instead colored the spectra [12]. The acoustic walls were shown to possess complex radiation characteristics, and the resonances can contribute to excessive reverberation times.

Lähdeoja *et al.* presented the measurements of a flat panel speaker constructed of plywood [10]. They measured the plywood, intended as a scenographic element to be viewed and heard from different directions, and designed a finite impulse response (FIR) equalizer (EQ) using the inverse Fourier transform. Due to their application, they opted for an averaged multi-point design procedure that resulted in an acceptable compromise regarding the magnitude responses in different directions. The measurements showed that without the equalization the audio output was heavily dependent on the acoustic properties of the radiation surface and its modes, which led to a heavily colored spectrum with emphasized resonant modes [10]. Other potential problems are the lack of perceived bass, a blurry bass response, and a reduced dynamic range.

Cecchi *et al.* studied the effects of equalization on sound transducers installed on existing surfaces, such as walls, ceilings, or swimming pool walls [5]. They measured different vibrating surfaces in various environments and noted that the resulting magnitude response is not flat in general. Thus, a multi-point equalization procedure was applied to enhance the sound quality. Both objective and subjective tests indicated positive effects: resonances were reduced, the magnitude response became flatter, and the overall audio quality was improved [5].

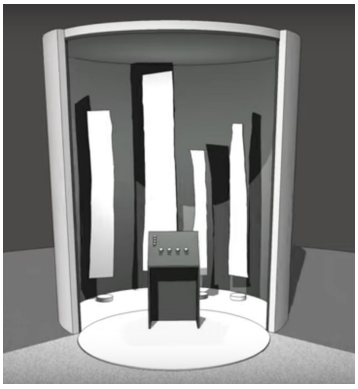


Figure 1. Sketch of the tree orchestra installation at the Finnish Science Center Heureka (used with permission). The control panel in the center allows the user to adjust the volume of each speaker and select the program material.

Figure 2. Tree orchestra installation during the mixing of the music program. From left to right: spruce, maple, goat willow (*Salix caprea*), and apple.

This paper presents measurements of a "tree orchestra" installation. This project is a commission from Heureka, a science centre in Vantaa, Finland. As part of their 2020 exhibition on wood called Wild Wild Wood, Heureka sought to have an installation showcasing the music-related acoustic properties of wood in an engaging and aesthetically attractive manner. In response to Heureka's request, our team at the Aalto Acoustics Lab and at the University of the Arts Helsinki, joined by luthier Juhana Nyrhinen, elaborated a design idea for a tree orchestra comprising four wooden panels equipped with structure-borne sound drivers. In order to allow for easy comparison between the different panels, the installation contains audience interaction in the form of playback controls. The work builds on previous research on artistic use of audio-rate vibration in solids [10], involving several sound art installations [14]. As the wood panels have a highly colored response, the use of an EQ is considered helpful to improve the overall sound quality. However, the goal is to retain the characteristic wood resonances.

This paper is organized as follows. Section 2 describes the design principles and the construction of the four wood-panel loudspeakers. Section 3 focuses on the acoustic measurements and the EQ design for the wood-panel speakers, comparing two different equalization strategies: multi-point and single-point. Section 4 analyzes the results and shows how the panels' sound quality was improved using EQs. Section 5 concludes the paper.

2. DESIGN OF THE TREE ORCHESTRA

The installation is composed of four wood panels cut in longitudinal sections directly from the trees, exposing the tree's internal structure as well as its contour. The core idea is to play original composed music through the panels, one instrument per panel, following the metaphor of a "tree orchestra". A sketch of the installation is shown in Fig. 1.

One of the authors (Otso Lähdeoja) composed a set of four musical pieces and recorded them with an instrumental quartet comporting a cello, violin, clarinet, and flute. A central design guideline was to use local Finnish wood traditionally used in lutherie. The overall design targeted strong visual appeal and character combined with optimized audio quality.

The wood and musical instruments were assigned as follows, with the approximate panel size in parenthesis (height, width, depth): Spruce (250 cm × 40 cm × 2 cm) – cello; maple (202 cm × 30 cm × 2 cm) – clarinet; goat willow (183 cm × 22 cm × 1 cm) – violin; apple (161 cm × 16 cm × 1 cm) – flute. The choice of the instrumental ensemble was made on aesthetic grounds, aiming for a light, acoustic ensemble sound. The assignment of the instruments to the different wood panels was decided upon testing how the unprocessed audio recordings translated through each panel. Figure 2 shows the tree orchestra setup during the final mixing of the musical pieces.

The panels are equipped with audio transducers (structure-borne sound driver) for sound output. Each panel has one Tectonic Audio Labs TEAX32C30-4/B transducer [1] and one Fischer Amps Bass Pump 3 [2]. The smaller actuator, TEAX32C30-4/B, has a reported frequency range from 100 Hz to 20 kHz. The bass drivers reportedly respond between 5 and 200 Hz. Thus, the two transducers implement a built-in crossover, and they are treated as one loudspeaker unit in this study.

The low-frequency drivers were placed in the lower half of the panels and the treble drivers were placed in the upper half. Generally, the actuators were placed slightly off the center line to reduce symmetric modes. The distance between the two actuators also affects the generated vibrations in the panels, and in this study, the distance was maximized while at the same time ensuring that they are not placed too close to the top or bottom of the panels. Finally, the exact location of the drivers on each panel was determined by ear due to the non-standard nature of the panels.

The transducers are driven by two Audac EPA104 D-

[1] https://www.tectonicaudiolabs.com/wp-content/uploads/2019/04/T-DS-TEAX32C30-4B_Rev-1.1.pdf

[2] https://www.fischer-amps.de/drum_section.html#article-253

(a) Front (b) Rear

Figure 3. Placement of the maple panel loudspeaker in the anechoic chamber: (a) the front and (b) the rear of the loudspeaker, showing the bass actuator below the middle and the treble actuator near the top highlighted with arrows.

class power amplifiers [3]. The audio is played back with a custom-made four-track audio-player program, incorporating an interface allowing the public to mix between tracks and their respective instruments. The audience interaction rationale is to provide the public with an opportunity to engage in active participation by navigating within the musical composition and the instrumentation of the installation as well as to allow for a careful listening of each wood panel's specific sonic qualities.

3. MEASUREMENTS AND EQUALIZER DESIGN

3.1 Measurement Set-up

All the measurements were conducted in the large anechoic chamber in Aalto University's Acoustics Lab. The wood-panel loudspeakers were hung in the middle of the chamber one at a time, as shown in Fig. 3, attached loosely to the net floor in order to keep them still and facing the microphones. This is also the way the wood-panel loudspeakers are positioned in the tree orchestra installation in the science center, as shown in Fig. 1. Furthermore, should the wood-panel loudspeakers be positioned directly in contact with the floor or other hard surface, their frequency response would change due to the altered radiation properties, and their vibrations could be transmitted to the supporting structures. The aim was to create an audio system in which the sound radiates directly from the wood panels and not through any of the supporting structures.

All wood-panel speakers were positioned with the bottom edge about 15 cm above the net floor of the anechoic chamber. The height of the panels varied from 161 to 250 cm, and thus, the midpoint of the speakers in this set-up varied from 99 to 140 cm. Figure 3 shows the placement of the maple-panel speaker during the response measurement.

[3] https://audac.eu/Products/d/epa104---quad-channel-class-d-amplifier-4-x-100w---crossover

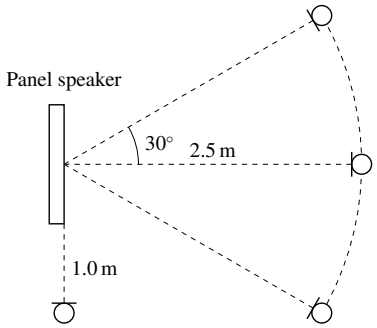


Figure 4. Locations of three microphones in the front sector and one on the side of the panel.

The response of the wood-panel speakers were measured using five G.R.A.S Type 46AF 1/2-inch, free-field microphones. Figure 4 illustrates the placement of the microphones. Four microphones were placed at a height of 1.6 m (those shown in Fig. 4) and one at a height of 1 m, corresponding to an assumed ear height for an adult and a child, respectively. School children and families with small children form a large portion of the visitors to the science center. The four microphones were positioned 2.5 m away from the loudspeaker at angles $-30°$, $0°$, and $30°$. These values were selected to approximate the audience position in the installation. The "child" microphone was placed at the 0-degree angle below the "adult" microphone. One microphone was placed at the side of the loudspeaker, at an angle of $90°$, in order to verify how a dipole loudspeaker of this type without a baffle radiates sound to the side.

The acoustic measurements were conducted by using a 5-second logarithmic sine sweep [15]. The actuators—two per wood-panel speaker, as seen in Fig. 3(b)—were measured one at a time, as well as simultaneously. The playback level of the sweeps, i.e., the amplifier gain, was kept constant throughout the measurements. Thus, all level differences in the responses are caused by the different types and sizes of the wood panels. The equalization was designed based on these measurements (see Sec. 3.2), after which the equalized wood-panel loudspeakers were re-measured to confirm that the magnitude response of the loudspeaker actually changed as was intended (see Sec. 4).

3.2 Single-Point and Multi-Point Equalizer Design

Based on the first set of measurements, a cascade graphic equalizer (GEQ) was defined to flatten the overall magnitude response of each wood-panel loudspeaker. Since the response of each wood type was measured with multiple microphones, two different procedures were tested: one based on averaged responses of several microphones (multi-point) and another based on a single response (single-point). First, the magnitude responses were smoothed using a one-sixth-octave averaging window in order to decrease the effect of sudden changes in the mag-

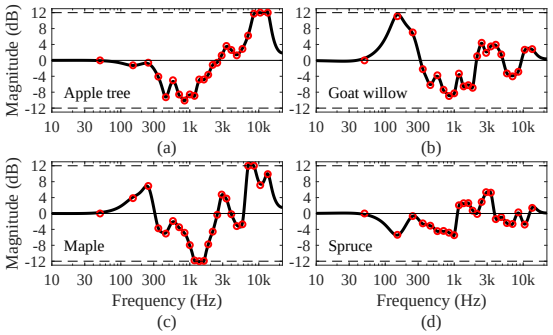


Figure 5. Single-point Bark GEQ responses for the (a) apple tree, (b) goat willow, (c) maple, and (d) spruce panel speakers. The red circles are the command gains, or the estimated corrections needed at each Bark band. The lowest center frequency is 50 Hz and the highest is 13.5 kHz.

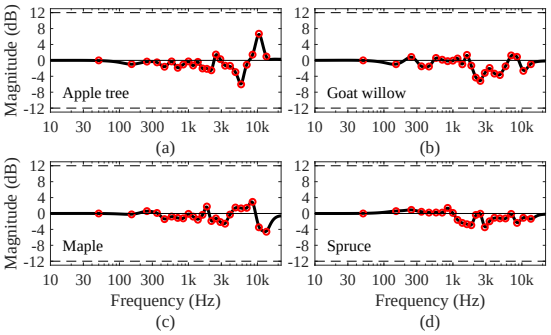


Figure 6. Multi-point EQ responses for the wood-panel loudspeakers, cf. Fig. 5.

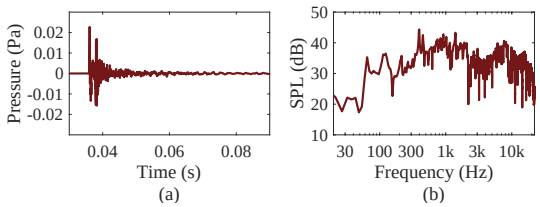


Figure 7. Measured impulse response and the corresponding magnitude response (without smoothing) of the goat willow panel at microphone 1 in front of the speaker.

nitude response of the EQ. Next, the responses measured with the four microphones in front of the wood-panel loudspeakers were averaged. This was the baseline signal for the multi-point EQ design procedure. For the second procedure, the smoothed response of microphone 1 (distance 2.5 m, height 1.6 m) was used as the baseline.

The EQ design requires only the command gains at the band center frequencies. A recent neural-network controlled Bark-band GEQ [16] comprising 24 bands was used in this project. It was chosen due to its accuracy and low computational load. The command gains for all bands were estimated as follows. The target response was set to be constant (flat) in the passband of the wood-panel speakers, and so the smoothed baseline magnitude responses specified above were additively inverted to determine the gain required for a flat response. That is, the mean value of the baseline response was first subtracted from the baseline response in order to bring it to around 0 dB. Next, the dB values at the Bark center frequencies were picked, and their additive inverse values were stored as command gains. In order to avoid excessive gains, a $\pm$12-dB limit was set for the maximum gain values. Furthermore, the gain for the first band (50 Hz) was always set to zero so as not to overload the bass transducer. This way, two sets of command gains were obtained from the two different design procedures (single-point and multi-point).

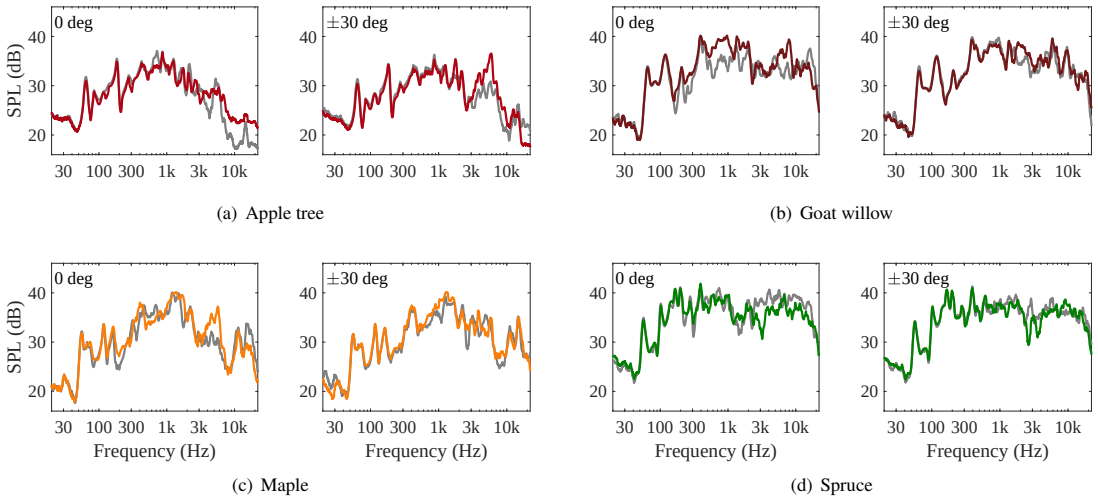
The EQ design itself was based on a neural network: the command gains were fed to a four-layer neural network, which selected the optimal band-filter gains by accounting for the interaction between adjacent Bark-band filters [16]. These filter gains were used to design a second-order filter for each Bark band of the GEQ, as described in [16]. The resulting EQ responses for each wood type are shown in Fig. 5 for the single-microphone design method and in Fig. 6 for the multi-point method. The required EQs for the different wood-panel loudspeakers differ from each other, demonstrating the different acoustic behavior of each wood type and panel size. The comparison of Figs. 5 and 6 also reveals that the two design methods produce completely different EQ curves.

The most striking difference between Figs. 5 and 6 is in the range of correction: almost all large peaks and dips are missing in the latter. The gains in Fig. 6 mainly range from approximately -6 dB to 7 dB, whereas in Fig. 5, practically the entire range of $\pm$12 dB is used except in the case of the spruce-panel speaker. The explanation for this is that the responses of the wood-panel loudspeakers vary greatly as a function of direction (see the figures in Sec. 4), and thus, when the EQ is designed based on averaged results from different microphones, the effect of extreme values dissolves, and the resulting EQ has little effect.

It was quickly noticed that the EQs in Fig. 5 produce better results especially in the front direction (this is analyzed further in Sec. 4). This is the most important direction for the wood-panel loudspeakers, since they will be exhibited in a rather small enclosure, as shown in Fig. 1, to avoid sound from radiating all over the exhibition space. Thus, only a few listeners can enjoy the speakers at one time, which allows us to concentrate on improving only the sound radiating directly in front of the speakers. This substantiates the choice of the EQs in Fig. 5. Their effect was tested for each wood-panel loudspeaker by listening to the combined effect of the actuators and the EQs to ensure no audible artifacts emerged due to the equalization. Finally, the effect of the EQs was measured to verify their behavior. These results are presented next.

4. RESULTS

This section reports the results for the two sets of measurements: the wood-panel loudspeakers without and with



(a) Apple tree (b) Goat willow

(c) Maple (d) Spruce

Figure 8. Unequalized responses for all wood types. For each (a)–(d), the left figure is for 0° at height 1.6 m (color) and 1.0 m (gray) whereas the right one is for ±30° (color and gray, respectively) at height 1.6 m.

the equalization, respectively. As explained in Sec. 2, the loudspeakers are not optimized electroacoustically, i.e., the two actuators are fed with identical signals from two different amplifier output channels without crossover filters or time-alignment. Thus, we consider the combined effects of the two actuators and the wood itself by having one EQ for both channels and by not considering the separate responses measured from one actuator at a time. Furthermore, we acknowledge the possible time differences between the two actuators seen in the measured impulse responses (see an example in Fig. 7(a)), but we ignore them. These time differences range from couple milliseconds to less than ten milliseconds.

Figure 7 shows an example impulse response and a non-smoothed magnitude response obtained with the microphone in front of the speaker at height 1.6 m. Two separate main spikes are seen in the impulse response in Fig. 7(a). This is caused by a time difference between the two actuators, the amount of time each actuator requires to excite the wood, and the difference in distance between the two actuator locations and the microphone. The impulse response is also longer than that of a typical loudspeaker.

The corresponding magnitude response is shown in Fig. 7(b), demonstrating the non-ideal unequalized response. Many peaks are seen corresponding to resonances as well as the general unevenness of the response. Due to the roughness of the obtained responses, in the following, the magnitude response curves are presented after a one-sixth-octave smoothing for better clarity. The GEQ with 24 bands is unable to fully flatten the magnitude response. Thus, important information about the overall shape of the magnitude response is not lost by applying the smoothing.

4.1 Unequalized Responses

The unequalized magnitude responses for each wood-panel loudspeaker are shown in Figs. 8(a)–8(d) for the two microphones at 0° and two others at ±30° (the figures are color-coded across the paper). All frontal responses (0°) in Fig. 8 contain resonances in the upper bass range (between 40 Hz and 200 Hz), which give the wood panels a boxy sound quality [17]. These resonances occur in all curves in Fig. 8 (both at 0° and ±30°) indicating that they arise from properties having quite wide radiation patterns.

From the 0-degree plots in Fig. 8, one sees that the wood-panel loudspeakers, apart from the spruce, mainly radiate middle frequencies (around 1 kHz) well, but the low and high frequencies less effectively. The spruce-panel loudspeaker possesses the flattest overall response aside from and the wide dip around 2 kHz. The flatness is also visible in the EQ curves in Fig. 5(d), where the EQ for the spruce contains the smallest gains, i.e., its overall shape is closest to a flat line. In addition, the responses from the adult and child microphones differ from each other for all wood types: Figs. 8(a)–8(c) show that there is more energy at the higher microphone location, probably due to the small treble actuator being placed high on the wood-panel speakers and being unable to excite the entire panel of wood to radiate sound thus resulting in a high directivity. The spruce-panel speaker differs from the others in this regard.

When observing the ±30-degree plots in Fig. 8, the wood-panel loudspeakers are seen to be asymmetric radiators, since the two curves differ from one another for all wood types. The low-frequency resonances, however, are mostly identical for ±30°. As mentioned, these resonances are also similar here as in the 0-degree figures for the corresponding wood type, whereas the other frequency regions differ between 0-degree and ±30-degree responses. The maple and the spruce panels have the widest radiation patterns, as seen in Fig. 8(c) and Fig. 8(d), since the ±30-degree responses resemble most the ones at 0°. This is logical since these two wood panels are the largest ones. The spruce-panel speaker radiates the flattest magnitude

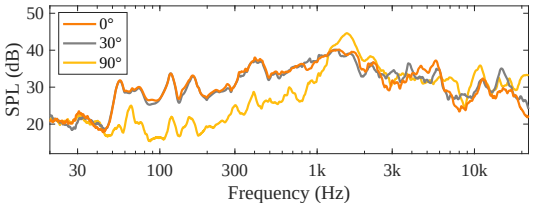


Figure 9. Directivity of a wood-panel loudspeaker (maple).

response in the ±30-degree directions.

4.2 Directivity

Figure 9 presents the measured directivity of the maple panel loudspeaker by showing the magnitude responses in the directions of 0°, 30°, and 90°. The response measured at 90° differs drastically compared to the other two directions, especially below 1 kHz. The attenuation of low frequencies is caused by the cancellation of sound waves radiating forward and backward from the wood-panel speaker in opposite phase. This acoustic short-circuiting is a typical behavior of a dipole radiator. Otherwise, the magnitude levels are similar, especially when observing the curves measured at 0° and 30°. Thus, one can conclude that even though the wood-panel loudspeakers do not radiate ideally due to asymmetries in the wood, the magnitude responses are good in the frontal sector. This analysis applies to all measured wood panels, although only one of them is shown here as an example.

4.3 Equalized Main Frontal Response

After the EQ design based on the initial acoustic measurements, a second set of measurements was conducted to verify that the equalized panels performed as predicted. Here, the results of the second measurement are compared with the first ones for the different panels and microphone locations. Figure 10 shows the effect of the equalization by presenting both the unequalized and the equalized magnitude response for each loudspeaker for the "adult" microphone (0°, height 1.6 m). Note that each curve is normalized by setting the level at 1 kHz to 0 dB.

Figure 10 demonstrates that the EQs produce the desired result for each panel. The magnitude response of the apple-panel speaker originally contained almost 14 dB of variation between 60 Hz and 20 kHz (the −6-dB corner frequencies for the equalized response), whereas after equalization the largest deviations from the normalized level are, naturally, 6 dB for the same frequency range and only approximately 2 dB between 300 Hz and 10 kHz. The resonances of the wooden panel are still visible in the response, but the general level is flat.

The goat-willow-panel loudspeaker response originally contained larger deviations at low frequencies than the apple tree one, but the response at high frequencies behaved better, as shown in Fig. 10. After the equalization, a relatively flat response is observed: before the equalization the largest deviations were approximately 12 dB between 60 Hz and 20 kHz (again, the −6-dB corner frequencies),

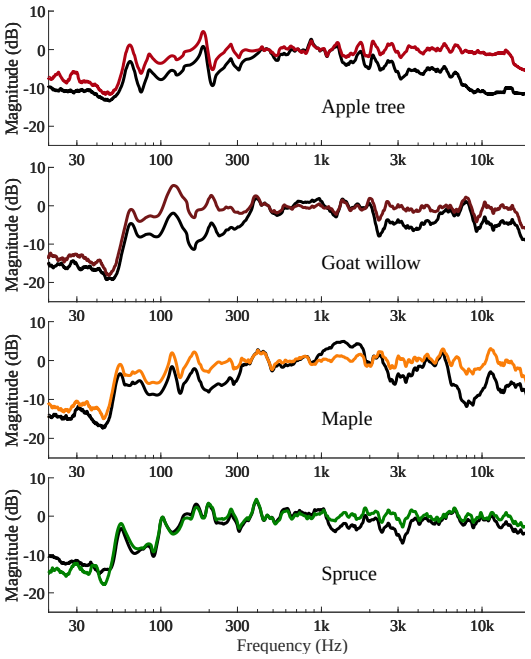


Figure 10. Main frontal magnitude responses of all wood panels before (black line) and after (color line) equalization, one-sixth-octave smoothed.

whereas after the equalization the value is approximately 4 dB for most of the frequency bands. A resonance around 120 Hz is seen to be boosted above the reference level. Not much could have been changed in the EQ response due to the selected frequency division. Furthermore, the stated resonance peak did not affect the sound negatively when listening to the equalized speaker.

The unequalized response of the maple-panel speaker in Fig. 10 is similar to that of the apple-tree-panel in that it has a wide peak around 1 kHz. The maple has, however, also other wide peaks, and the largest deviations from the reference level equal approximately 10 dB between the −6-dB corner frequencies of 50 Hz and 19 kHz. After the equalization, the response is much flatter, with the largest deviation being approximately 6 dB at low frequencies and no more than 3 dB at high frequencies.

The unequalized magnitude response of the spruce panel in Fig. 10 is the flattest of the four. Still, the magnitude differs from the normalized level by as much as about 10 dB between 50 Hz and 21 kHz (i.e., the −6-dB corner frequencies of the equalized response). After the equalization, the largest deviation is approximately 9 dB below 100 Hz and less than approximately 4 dB above that. For the equalized spruce-panel loudspeaker, the lowest resonance peak of the magnitude response deviates the most from the rest of the response, so nothing could be done about it with the selected EQ. Between 150 Hz and 400 Hz, three peaks are preserved after the equalization, i.e., the EQ neither boosts nor attenuates them due to the wide bandwidth of the EQ filters relative to the said peaks.

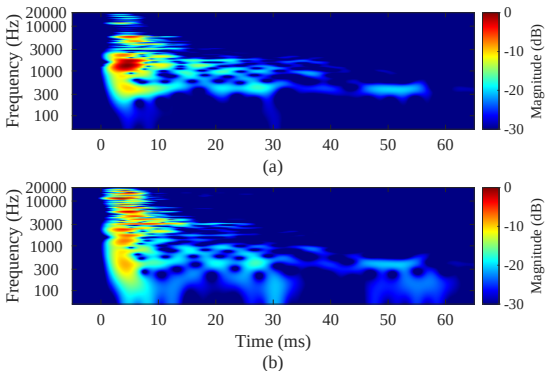


Figure 11. Spectrograms of the (a) unequalized and (b) equalized maple-panel speaker's response measured at the front microphone.

4.4 Time-Frequency Analysis of Measured Responses

Finally, spectrograms of the impulse responses on a logarithmic frequency scale were computed to analyze the wood-panel loudspeakers and the effects of equalization. Figure 11(a) shows the spectrogram of the unequalized response of the maple panel, which is computed similarly as in [18]. We use a 10-ms long Blackman window, a hop size of 1 sample, and 256 logarithmically-spaced frequencies to evaluate the discrete-time Fourier transform. The most notable property here is the lack of low frequencies and the temporal spreading of the response at multiple frequencies. Additionally, the main impulse around 5 ms, which corresponds to the magnitude response in Fig. 10, is not of the same color, i.e., the magnitude response is not flat.

The equalized response in Fig. 11(b) contains similar ringing properties to the unequalized one, i.e., the equalization does not cancel the resonances of the wood panels. Now, due to the EQ, the main impulse is closer to having a constant color, i.e., the magnitude is flatter. At the same time, however, the low frequencies introduced by the EQ spread heavily in time. Thus, the log-spectrograms show that it is important to consider both the temporal as well as the frequency-domain properties of loudspeakers. The spectrograms also exemplify the limitations of the minimum-phase EQs in that they cannot repair the non-minimum-phase problems in sound systems [19].

In this application, where non-ideal wood-panel loudspeakers are used, the aim was not to achieve perfection, but improvements to the initial situation. This goal was achieved, as confirmed by informal listening of the panels. The listening comprised the composer listening to the equalized wood-panel speakers and carefully verifying the best panels for each instrument. The spruce-panel speaker produced the most pleasing sound, which is not surprising considering spruce is used in musical instruments.

The final verification occurred when the EQs were applied to the music tracks. The composer noted an improved sound quality and increased clarity for every speaker. In addition, the original characteristics of the wood were not fully lost, and the spatial sound image remained exciting.

4.5 Equalized Responses in Other Directions

The responses measured with the front microphone were discussed above, since that microphone was used for the EQ design. It is, however, interesting to consider the effects of the EQs at other measurement points as well. The responses at 30° at height 1.6 m and 0° at height 1 m are shown in Fig. 12 for every wood type. The responses are offset to improve clarity. Only responses from the $+30$-degree microphone are considered here, but the -30-degree microphone signals show similar trends while also containing some differences in the magnitude responses, as suggested by Fig. 8.

Figure 12(a) shows the magnitude responses at 30° after equalization for apple tree, goat willow, maple, and spruce. Comparing these to the equalized responses in Fig. 10, we notice the following. For the apple tree, the response has a similar flat shape aside from a small level difference and the wide boosted peak around 6 kHz.

The goat-willow-panel speaker, on the other hand, is flatter overall, and when compared to the front-microphone-response, the response at 30° in Fig. 12(a) is similar with small level differences and slightly changed peak structure. The deviations from a flat curve, however, do not grow much. The response of the maple-panel speaker at 30° resembles its frontal response. The overall response is a flat one with the resonance peaks moving around without their relative levels changing much. Figure 12(a) shows that the spruce-panel loudspeaker radiates a flat response in the 30-degree directions, resembling the 0-degree response with some resonance dips at different frequencies and depths.

Finally, the responses from the "child" microphone, i.e., the microphone below the front microphone at height 1.0 m, are analyzed. Overall, each response in Fig. 12(b) resembles the corresponding equalized magnitude response in Fig. 10, but differs at high frequencies: apple tree, goat willow, and maple radiate less energy in the child listening position, whereas spruce radiates more energy. In addition, there is a wide dip of about 5 dB at about 700 Hz in the equalized goat-willow-panel response in Fig. 10. Thus, although the magnitude responses are not as flat as in the "adult" microphone, children receive an improved sound as a result of the equalization.

Additionally, the responses from the side microphone (90-degree direction) were also analyzed (not shown), but the EQs had little effect on them: the responses are still bad due to destructive interference, and they are omitted here.

5. CONCLUSIONS

This paper presented a tree orchestra installation consisting of four sound-emitting wood panels. The wood panels were measured and equalized to have overall flatter magnitude responses. The aim was not to suppress the modes and resonances of the wood panels, but to reduce the coloration while still retaining the original reverberant characteristics of the panels. Hence, a Bark-band GEQ was utilized. The single-point equalization approach was found to be suitable for this application. The equalized wood-panel speakers were also measured to verify that the actuators

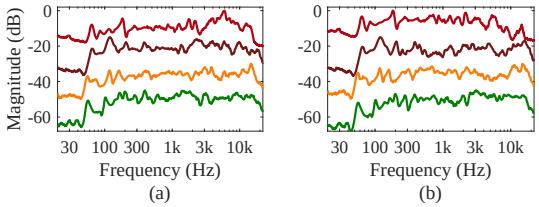


Figure 12. Effect of the EQs in different directions: (a) +30°, 1.6 m height and (b) 0°, 1.0 m height. From top to bottom: apple, goat willow, maple, spruce. The curves are normalized and offset by −15 dB from each other.

are capable of reproducing the enhanced sound without unwanted artifacts. Example anechoic recordings of the unequalized and equalized wood panels are available online at `http://research.spa.aalto.fi/publications/papers/smc2021-tree-orchestra/`.

Acknowledgments

This work was funded in part by the Aalto ELEC Doctoral School and by the Finnish Science Center Heureka. This research is part of the "Nordic Sound and Music Computing Network—NordicSMC," NordForsk project no. 86892. Otso Lähdeoja's work has been funded by the Academy of Finland research project grant no. 316072, "Intersubjectivity in Music; the Perspective of Technological Mediation." Luthier Juhana Nyrhinen has crafted the tree orchestra's wood panels. The authors would like to thank the following people at Heureka: Tapio Koivu and Mikko Myllykoski for the initial idea, Meiju Lampinen for production management, Mikko Kauhanen for providing Figure 1, and Matti Viirimäki for transporting the wood-panel speakers. Thank you to Luis Costa for proofreading the manuscript.

6. REFERENCES

[1] N. Harris and M. Hawksford, "The distributed-mode loudspeaker (DML) as a broad-band acoustic radiator," in *Proc. AES 103rd Conv.*, New York, NY, Sep. 1997.

[2] L. Hörchens and D. de Vries, "Comparison of measurement methods for the equalization of loudspeaker panels based on bending wave radiation," in *Proc. AES 130th Conv.*, London, UK, May 2011.

[3] A. Watson, "Unobtrusive audio: Current trends and design considerations," in *Proc. AES 21st UK Conf.: Audio at Home*, Cambridge, UK, Apr. 2006.

[4] G. Wersényi, "Evaluation of vibrating sound transducers with glass membrane based on measurements and numerical simulations," in *Proc. AES 132nd Conv.*, Budapest, Hungary, Apr. 2012.

[5] S. Cecchi, A. Terenzi, F. Piazza, and F. Bettarelli, "Applying sound equalization to vibrating sound transducers mounted on rigid panels," in *Proc. AES 147th Conv.*, New York, NY, USA, Oct. 2019.

[6] C. Sowden and F. J. Ampel, "Planar loudspeaker technology – Applications in the real world," in *Proc. AES 96th Conv.*, Amsterdam, The Netherlands, Feb. 1994.

[7] O. Mal, M. Novotný, B. Verbeeren, and N. Harris, "A novel glass laminated structure for flat panel loudspeakers," in *Proc. AES 124th Conv.*, Amsterdam, The Netherlands, May 2008.

[8] O. Lähdeoja and L. Reboursière, "Augmented window: Structure-borne sound drivers for sound-emitting solid objects and surfaces," *QPSR of the* NUMEDIART *Research Program*, vol. 4, no. 4, pp. 83–85, Dec. 2011.

[9] "Whispering Window Technical Specification," http://media.feonic.com/downloads/specs/Feonic%20Whispering%20Window%20Spec%20January%202015.pdf, 2013, accessed: 2021-05-26.

[10] O. Lähdeoja, A. Haapaniemi, and V. Välimäki, "Sonic scenography—Equalized structure-borne sound for aurally active set design," in *Proc. Int. Comput. Music Conf. (ICMC)/Sound and Music Computing Conf. (SMC)*, Athens, Greece, Sep. 2014, pp. 1725–1730.

[11] B. Pueo, J. Escolano, J. J. López, and G. Ramos, "A note on the filtering equalization in large multiactuator panels," in *Proc. 17th European Signal Processing Conf. (EUSIPCO)*, Glasgow, Scotland, Aug. 2009.

[12] G. Berndtsson, "Acoustical properties of wooden loudspeakers used in an artificial reverberation system," *Appl. Acoust.*, vol. 44, no. 1, pp. 7–23, 1995.

[13] G. Berndtsson and A. Krokstad, "A room acoustic experiment with an artificial reverberation system using wooden loudspeakers," *Acta Acustica*, vol. 2, no. 1, pp. 37–48, 1994.

[14] O. Lähdeoja, "Composing the context: Considerations on materially mediated electronic musicianship," *Organised Sound*, vol. 23, no. 1, pp. 61–70, Apr. 2018.

[15] A. Farina, "Simultaneous measurement of impulse response and distortion with a swept-sine technique," in *Proc. AES 108th Conv.*, Paris, France, Feb. 2000.

[16] J. Rämö, J. Liski, and V. Välimäki, "Third-octave and Bark graphic-equalizer design with symmetric band filters," *Appl. Sci.*, vol. 10, no. 4, pp. 1–22, Feb. 2020.

[17] T. H. Pedersen and N. Zacharov, "The development of a sound wheel for reproduced sound," in *Proc. AES 138th Conv.*, Warsaw, Poland, May 2015.

[18] V. Välimäki, J. S. Abel, and J. O. Smith, "Spectral delay filters," *J. Audio Eng. Soc.*, vol. 57, no. 7–8, pp. 521–531, Jul./Aug. 2009.

[19] S. P. Lipshitz, M. Pocock, and J. Vanderkooy, "On the audibility of midrange phase distortion in audio systems," *J. Audio Eng. Soc.*, vol. 30, no. 9, pp. 580–595, Sep. 1982.

INTRODUCING FINITE DIFFERENCE SCHEMES SYNTHESIS IN FAUST: A CELLULAR AUTOMATA APPROACH

Riccardo RUSSO (rrusso19@student.aau.dk) [1], **Stefania SERAFIN**[1], **Romain MICHON**[2,3], **Yann ORLAREY**[2], and **Stéphane LETZ**[2]

[1] **Aalborg University**, Copenhagen, Denmark
[2] **GRAME-CNCM**, Lyon, France
[3] *CCRMA*, **Stanford University**, USA

ABSTRACT

In this paper we propose a technique for formalizing Finite Difference Schemes (FDSs) physical models in the Faust programming language. Faust libraries already allow for the implementation of several kinds of physical modeling techniques; however, to our knowledge, FDSs have never been integrated into this language. In fact, their implementation in imperative programming languages is typically achieved using data structures, which are not available in Faust. First, a method for coding FDSs in a functional programming way is introduced, starting from previous works on mass-interaction models. Then, we draw a connection between FDSs and cellular automata, and exploit it for building a library that eases the implementation of FDS synthesis in Faust.

1. INTRODUCTION

Physical modeling has quite a long history in the field of sound synthesis. Over the years, many different techniques have been proposed [1], such as digital waveguides [2], mass-interaction models [3], modal synthesis [4] or finite difference schemes [5]. Even though sometimes more computationally demanding and difficult to control than other synthesis methods [6], physical modeling techniques offer many advantages. Indeed, creating a model of a vibrating system provides full control on its properties and, as a consequence, the output sound. These approaches theoretically allow us to synthesize natural and realistic sounds, tunable in every detail.

Faust [7] is a high-level, domain-specific, functional programming language, with a strong focus on the development of digital signal processing algorithms for sound and music. Faust code can be compiled with the Faust compiler, in order to directly generate standalone audio applications or plugins in different formats, or translate it to other languages such as C++, Java and others. The compiler includes options for automatically applying optimizations to the generated code. This feature is extremely useful for physical modeling synthesis, which requires fast code to perform big amounts of computation in real time.

1.1 Physical Modeling in Faust

Given the features stated above, its unique syntax, and the wide range of functions already available [8], Faust has been extensively used for implementing physical models. Smith [9] was among the firsts to exploit it in the context of physical modeling, by implementing simulations of a virtual electric guitar and various audio effects with digital waveguides. The latter, along with modal synthesis techniques, were also employed in the context of the Faust-STK [10], a collection of physical models based on some of the algorithms in the Synthesis ToolKit [11]. The Faust Physical Modeling Library was also recently implemented [12]. It contains models of various instrument parts that can be assembled together, and it introduces a new bi-directional algebra allowing for the implementation of coupling between the modules, at the cost of adding a one-sample delay. In addition, it formalizes a way to generate custom instrument parts by using `mesh2faust`, [1] a tool that performs finite element analysis on a 3D model and automatically generates a modal physical model.

In addition to modal and waveguide synthesis, mass-interaction physical models were also implemented in Faust: this technique consists of modeling physical systems in the form of lumped mass-spring networks [3]. Faust for mass-interaction was first explored by Berdahl and Smith with *Synth-A-Modeler* [13], a tool providing a high-level graphical environment to generate physical models by using a combination of mass-interaction and digital waveguides. More recently, Leonard et al, extending Berdahl's work, introduced *mi_faust*. This project contains the scripter MIMS [14], a high-level, graphical or command line tool that can be used to describe a physical model and automatically generate Faust code. [2] Along with the scripter, *mi_faust* includes `mi.lib`, a Faust library that can be used to assemble mass-interaction models directly in Faust in a modular way.

As seen above, several works employed Faust for developing physical models; however, to the authors' knowledge, Finite Difference Schemes (FDS) synthesis has never been integrated in this programming language. The implementation of FDS models with imperative languages is

[1] `ccrma.stanford.edu/~rmichon/pmFaust`
[2] `mi-creative.eu/tool_MIMS-Online_V2.html`

typically achieved using data structures such as vectors and matrices; which are not available in Faust. This might be a characteristic that has discouraged users interested in developing FDSs from trying to use this language. As a matter of fact, before this work, it was probably easier to code a FDS model directly in a lower-level language such as C++ rather than doing it in Faust. As such, this paper has two purposes: the first one is to introduce a routine for the development of FDS physical models in Faust, without making use of arrays; the second one is to provide a tool that allows for an easier implementation of such models.

2. FINITE DIFFERENCE SCHEMES

FDS synthesis consists in developing a full mathematical description of the system at hand, usually by employing partial differential equations, and then discretising the mathematical model using finite-difference time-domain (FDTD) methods, thus obtaining a finite difference scheme. This technique requires more computational power than other physical modeling methods such as the ones described in the previous section, and is more prone to numerical dispersion than, for example, digital waveguides [15]. However, it allows for a better spatial accuracy if frequency-dependent losses and dispersion are present [16] and it is more flexible, as FDTD methods do not make any assumptions on the system's solution.

FDTD methods essentially work by performing a discretization of the partial derivative operators. To do that, first a sampling grid for space and time has to be defined, thus we can write: $t = nk$ and $x = lh$ where $n \in \mathbb{N}$ and $l \in \mathbb{Z}$. The numbers k and h are the sampling steps of the system for time and space respectively; they are not independent and are bonded through a stability condition, which depends on the system equations. Given a system of PDEs in space and time with one-dimensional solution $u(x,t)$, it is possible to define the discrete function u_l^n which approximates it using the sampling steps above. Having defined a time grid, the time difference operators can be written as:

$$\begin{gathered}\delta_{t+}u_l^n = \frac{u_l^{n+1} - u_l^n}{k} \quad \delta_{t-}u_l^n = \frac{u_l^n - u_l^{n-1}}{k} \\ \delta_{t\cdot}u_l^n = \frac{u_l^{n+1} - u_l^{n-1}}{2k}\end{gathered} \tag{1}$$

These are the *forward*, *backward* and *center* difference operators respectively, which approximate the partial derivative operators. By combining them it is possible to obtain the definition for the second-order time difference operator:

$$\delta_{tt}u_l^n := \delta_{t+}\delta_{t-}u_l^n = \frac{u_l^{n+1} - 2u_l^n + u_l^{n-1}}{k^2} \tag{2}$$

The first and second order spatial operators are obtained in a similar way.

2.1 Example 1: 1-D Wave Equation

The discretization of the 1-D wave equation represents the simplest possible FDS. In the continuous case we have:

$$\ddot{u}(x,t) = c^2 \frac{\partial^2}{\partial x^2} u(x,t) \tag{3}$$

A FDS is given by:

$$\delta_{tt}u_l^n = c^2 \delta_{xx} u_l^n \tag{4}$$

If we expand the operators we obtain:

$$u_l^{n+1} = 2\left(1 - \frac{c^2k^2}{h^2}\right)u_l^n - u_l^{n-1} + \frac{c^2k^2}{h^2}\left(u_{l+1}^n + u_{l-1}^n\right) \tag{5}$$

2.2 Example 2: 2-D Wave Equation

The equation above can be extended in multiple space dimensions:

$$\ddot{u}(\mathbf{x},t) = c^2 \nabla^2 u(\mathbf{x},t) \tag{6}$$

if $\mathbf{x} \in \mathbb{R}^2$ a FDS can be written as:

$$\delta_{tt}u_{l,m}^n = c^2(\delta_{xx} + \delta_{yy})u_{l,m}^n \tag{7}$$

where l, m are the grid indexes in the two space dimensions. If the medium is isotropic then the two space sampling steps are equal: $h_x = h_y := h$. The operator expansion then yields:

$$\begin{aligned}u_{l,m}^{n+1} &= 2\left(1 - 2\frac{c^2k^2}{h^2}\right)u_{l,m}^n - u_{l,m}^{n-1} + \\ &+ \frac{c^2k^2}{h^2}(u_{l+1,m}^n + u_{l-1,m}^n + u_{l,m+1} + u_{l,m-1})\end{aligned} \tag{8}$$

The space-time dependencies (stencils) of equations (5) and (8) are depicted in Fig. 1.

Both schemes are linear and explicit; in fact the only unknown is one future point, which depends on a linear combination of the states of some spatial side points, itself and its delayed version. For more details on finite difference schemes, refer to Bilbao [5].

3. FDS IN FAUST

In imperative programming languages, FDSs are typically implemented using vectors and matrices: each time step is represented by a matrix of dimension equal to the space dimension. For instance, in the 2-D wave equation case, three 2-D matrices would be needed, for time steps $n+1$, n, $n-1$. The time states would then be updated at audio rate by cycling between the elements and applying the mathematical equations. Since all this is not possible in Faust, a different method had to be developed.

3.1 From Mass-Interaction to FDS

As before mentioned, in *mi_faust*, Leonard et al. introduced a library that allows for the implementation of mass-interaction physical models in Faust [14]. The `mi.lib`

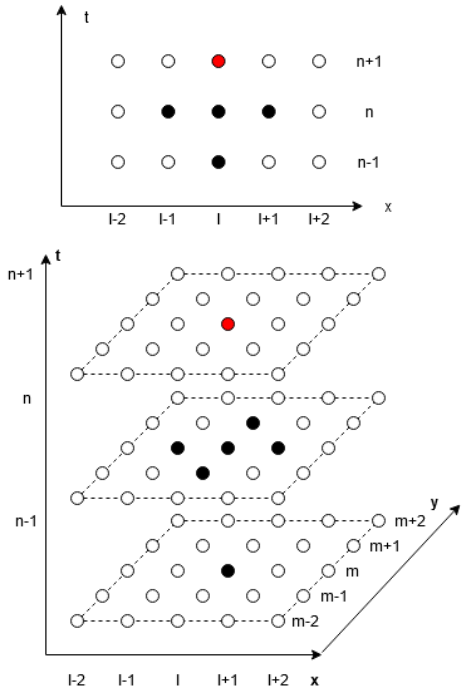


Figure 1. Stencils for the 1-D (top) and 2-D (bottom) wave equations. The red (top) point in each stencil is the unknown, the black points are the next points' states needed for the update equations

works this way: several *mass equation* blocks are stacked in parallel, followed in series by *spring-damper equation* ones; the number of these blocks is determined beforehand by the user when the mesh size is defined. The mass equations calculate the position of each mass given a force by discretising Newton's law: $\ddot{x} = F/m$. These are then fed into the dampened spring equations (from now on, *spring* will be used as a synonym for dampened spring), which calculates the force produced by each spring block depending on the position of the side masses, using Hooke's law and a linear damper equation: $F = -z(x_2 - x_1) - \sigma\frac{d}{dt}(x_2 - x_1)$, where z is the spring stiffness and σ the damping coefficient. The forces hereby calculated are then fed back into the mass equations, that output the new mass positions and so on. Each spring provides a force that only depends on two masses (the ones at the opposite sides of the spring) or one mass and a fixed point, which does not move. A mass position is then updated taking into account the forces coming from all the springs connected to it. The connections are previously defined by the user, depending on his mesh choices. Routing functions are used to bring the signals to the correct blocks.

If a uniform mesh is considered (namely, a mesh where each mass is connected with a spring to each neighbour, 2 in 1-D, 4 in 2-D, making up a string in 1-D or a regular matrix in 2-D), each mass receives a force from a constant number of springs, except the boundaries. In this case it is easy, by doing some algebraic calculations, to merge the mass and spring operations in one block. Given a time discretization as the one performed in section 2 and applying operator (2), Newton's law for a single mass at position 1 in the mesh becomes:

$$x_1^{n+1} = 2x_1^n - x_1^{n-1} + \frac{F_1^n k^2}{m_1} \tag{9}$$

This equation provides the horizontal position of m_1 at time step $n+1$. Using the backward operator (1), it is possible to discretize the force provided by the spring connecting m_1 to m_2, which will be applied to m_2:

$$\begin{aligned} F_{1\to2}^n = &-z_1(x_2^n - x_1^n) + \\ &- \frac{\sigma_1}{k}((x_2^n - x_2^{n-1}) - (x_1^n - x_1^{n-1})) \end{aligned} \tag{10}$$

And, for Newton's third law, the force applied to m_1 will be $F_{2\to1}^n = -F_{1\to2}^n$. If we consider a 1-D mesh, the total force applied to the single mass m_1 will be the sum of the forces provided by the two side springs: $F_1^n = F_{0\to1}^n + F_{2\to1}^n$. Therefore, it is possible to generalize equation (9) for a generic mass m_l not situated at boundaries:

$$\begin{aligned} x_l^{n+1} = &\, 2x_l^n - x_l^{n-1} + \frac{k^2}{m_l}\{-z_j(x_l^n - x_{l-1}^n) + \\ &- \frac{\sigma_j}{k}[(x_l^n - x_l^{n-1}) - (x_{l-1}^n - x_{l-1}^{n-1})] + \\ &+ z_{j+1}(x_{l+1}^n - x_l^n) + \\ &+ \frac{\sigma_{j+1}}{k}[(x_{l+1}^n - x_{l+1}^{n-1}) - (x_l^n - x_l^{n-1})]\} \end{aligned} \tag{11}$$

where z_j and σ_j are the stiffness and damping coefficient for the j-th spring.

If seen from another point of view, what we obtained here is the update equation for a linear, explicit FDS model. In fact, in equation (11) the future state (a horizontal position in this case) of a spatial point is given by a linear combination of the present and past states of itself and its *neighbours*. The fact that we started by considering masses and springs reflects here only in multiplications by the scalar coefficients m, z, and σ.

3.2 Faust Implementation

Having proved that there is a connection between mass-interaction and FDS, it is now possible to implement a FDS model in Faust, taking inspiration from the *mi_faust* approach. Using equation (11) we can merge the mass and spring blocks into one: now, each block will output its state, feed it back and receive in input the states signals from its neighbours and itself. The Faust syntax allows us to get the past versions of the states by simply using the delay operator `'`. The compiler automatically allocates the needed memory; therefore it is not necessary to implement multiple data structures for saving previous data, as it is in imperative languages. With this configuration we are also not limited to implement only equations (11), but whichever explicit update equation depending on a fixed number of neighbours' states. As an example, code listing 1 shows the Faust function for the 1-D wave equation (5):

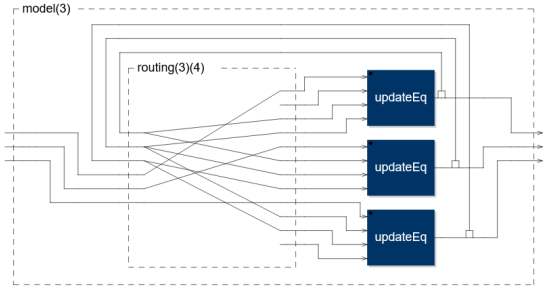


Figure 2. Block diagram for a 3-point-long 1-D wave equation scheme

```
lambda = c*k/h;
updateEq(fIn,u_w,u,u_e) =
   2*(1-lambda^2)*u-u'+lambda^2(u_w+u_e)+fIn;
```

Listing 1. 1-D wave update equation

This block takes as inputs the left and right neighbours' current states `u_w` and `u_e`, its own current state `u` and an external force `fIn`, and outputs the next state of the spatial point. Many blocks of this kind can be stacked in parallel to form meshes of the desired size (as done in listing 2): each one of these can be thought of as a single cell of the arrays that are usually implemented in imperative programming languages.

```
build1DScheme(nPoints)=par(i,nPoints,updateEq);
```

Listing 2. 1-D model builder function

As an example, Fig. 2 shows the Faust diagram for a 3-point-long 1-D wave equation scheme (the scheme was made very small in order to improve the image readability), coded by stacking in parallel the function in listing 1 three times. Here the signal paths described above are clearly visible. It can be noticed that the side blocks have an empty connection each; in fact, being at boundaries, there is no neighbour to take the signal from, so here they receive a zero value. It is possible to specify a block with a different equation that takes into account boundary conditions; if this is not done, as in this case, the zero signal automatically implies a clamped boundary condition. While the neighbour states are taken from the feedback loop, the force signals are open connections, these can be calculated outside the model and sent to the desired block with a selector function or an interpolator. This allows us to excite the mesh in the proper position.

The routing function is an essential part of the algorithm, and can be implemented using the Faust `route` primitive, which allows to drive the correct signals into the wanted spots in an optimised way. Code listing 3 shows the routing used for the algorithm in Fig. 2. Here `nPoints=3` is the number of blocks (points) stacked in parallel and `nInputs=4` is the number of inputs for each block. The primitive is designed so that the number of connections is 1-indexed: if a connection is numbered zero, or falls outside the maximum number of connections specified as an argument, a zero value is sent. This feature allows us to implement the "empty" connections for the boundaries shown above; the functions `F, W, C, E` take care of doing this.

```
routing(nPoints,nInputs) =
  route(nPoints+nPoints, nPoints*nInputs,
    par(x, nPoints, connections(x)))
with
{
  connections(x) =
    P(x)+nPoints, F(x),
    P(x), E(x-1),
    P(x), C(x),
    P(x), W(x+1);
  P(x)=x+1;
  F(x)=(1+0+(x*nInputs))*(x>=0)*(x<nPoints);
  W(x)=(1+1+(x*nInputs))*(x>=0)*(x<nPoints);
  C(x)=(1+2+(x*nInputs))*(x>=0)*(x<nPoints);
  E(x)=(1+3+(x*nInputs))*(x>=0)*(x<nPoints);
};
```

Listing 3. Routing function for 1-D wave equation.

The same algorithm structure can be employed for coding 2 or 3-D models. Listing 4 shows the Faust implementation for the 2-D wave update equation obtained in (8).

```
lambda = c*k/h;
updateEq(fIn,u_n,u_s,u,u_w,u_e) =
  2*(1-2*lambda^2)*u-u'+lambda^2*(u_e+u_w+u_n+
    u_s)+fIn;
```

Listing 4. 2-D wave update equation.

Again, many copies of this block can be stacked in parallel to form a 2-D mesh. Faust does not provide multi-dimensional structures such as matrices; therefore, this operation is not as straightforward as it was before. A nested for-loop can be simulated by nesting two par iterations, resulting in a piece of code that looks similar to what is used in imperative languages for parsing matrices:

```
build2DScheme(X,Y) = par(x,X,par(y,Y,updateEq));
```

Listing 5. 2-D model builder function

where `X` and `Y` are the total number of points in the x and y dimensions. Nevertheless, this algorithm will not form a 2-D structure; on the contrary, it will unroll it and build a 1-D block sequence with length `X*Y`, where consecutive rows are put one after the other. Hence, the code above is only useful to keep the double indexing convention.

Not having multi-dimensional structures is only a partial issue; in fact, with a proper routing, it is possible to drive the correct feedback signals into the right blocks. The routing function for the 2-D wave equation can be implemented as in listing 6:

```
routing2D(X, Y, nInputs) =
  route(X*Y*2, X*Y*nInputs,
    par(x, X, par(y, Y, connections(x,y))))
with
{
  connections(x,y) =
    P(x,y) + X*Y, F(x,y),
    P(x,y), S(x,y-1),
    P(x,y), N(x,y+1),
    P(x,y), C(x,y),
    P(x,y), E(x-1,y),
    P(x,y), W(x+1,y);
  P(x,y)=x*Y+y+1;
  F(x,y)=(1+0+(x*Y+y)*nInputs)*(x>=0)*(x<X)*(y
    >=0)*(y<Y);
  N(x,y)=(1+1+(x*Y+y)*nInputs)*(x>=0)*(x<X)*(y
    >=0)*(y<Y);
  S(x,y)=(1+2+(x*Y+y)*nInputs)*(x>=0)*(x<X)*(y
    >=0)*(y<Y);
  C(x,y)=(1+3+(x*Y+y)*nInputs)*(x>=0)*(x<X)*(y
    >=0)*(y<Y);
```

```
18   W(x,y)=(1+4+(x*Y+y)*nInputs)*(x>=0)*(x<X)*(y
       >=0)*(y<Y);
19   E(x,y)=(1+5+(x*Y+y)*nInputs)*(x>=0)*(x<X)*(y
       >=0)*(y<Y);
20 };
```

Listing 6. Routing function for 2-D wave equation.

Again, `X` and `Y` are the number of points in the two spatial dimensions, and `nInputs=6` is the number of inputs of each block.

Comparisons between these algorithms coded in Faust and the same versions in Matlab showed that the outcoming data was exactly the same; therefore it is possible to state that this method allows us to solve explicit finite difference schemes.

4. FDS LIBRARY

In the previous section, a method for implementing FDS physical models in Faust has been introduced and it was showed that what is usually achieved with array indexing can be equivalently accomplished in Faust by specifying signals routings. However, we can say without a doubt that coding FDSs this way is not as straightforward as it is in imperative languages. The main difficulty comes from the routing function, which can become very complicated for some models. In fact, the examples reported above are the simplest ones to implement, and things can become more intricate when a larger number of neighbours is needed. Moreover, these functions have to be re-written from scratch for each scheme. On the other hand, routing functions are not needed in languages where data structures are available, as they are replaced by explicit signal indexing. This makes Faust not competitive, at the moment, when it comes to implement FDS synthesis. For this reason, a second goal was set, consisting in the development of a library, `fds.lib`, allowing for a faster implementation of FDSs in Faust. Since this aims to be an introductory work and the intention was to give a coherent structure to the code, it was decided to only focus on linear and explicit schemes, leaving the implementation of other kind of simulations for future work. While it may seem a limitation, it has to be considered that linear schemes are sufficient for many cases of musical interest [5]; in fact, the majority of FDS models that run in real-time nowadays are of this kind, and Faust is specifically oriented towards the development of real time applications.

4.1 Cellular Automata

A cellular automaton (CA) is an algorithm that operates on a grid of cells, which can be in a finite number of states. For each cell, a set of cells is defined and called *neighbourhood*: at each time step t, the next state of a cell is determined by its present state and the state of its neighbours. The rule determining the new state is called *transition rule* and can be linear or nonlinear. The number of neighbours is defined by a coefficient r, called the neighbourhood *radius*; this indicates the number of cells at each side of the current cell that are taken into account. For instance, if $r = 1$ and the scheme is 1-D, the transition rule will depend on the current cell, one cell on the right and one on the left. Therefore, a neighbourhood is made of $(2r + 1)^D$ elements, where D represents the scheme dimension.

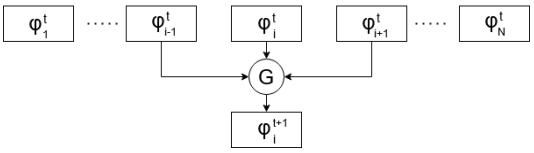


Figure 3. Scheme of a 1-D CA algorithm with transition rule G

If we consider a 2-D system, we can call the system's state at time t $\mathbf{\Phi}^t$; therefore the i-th, j-th single cell's state can be written as $\phi^t_{i,j}$. We can then define a transition rule G that brings $\mathbf{\Phi}^t$ to $\mathbf{\Phi}^{t+1}$ such that $G : \mathbf{\Phi}^t \mapsto \mathbf{\Phi}^{t+1}$. If G is linear we can write:

$$\phi^{t+1}_{i,j} = \sum_{\alpha=-r}^{r} \sum_{\beta=-r}^{r} a_{\alpha,\beta} \phi^t_{i+\alpha,j+\beta} \tag{12}$$

where $a_{\alpha,\beta}$ are the coefficients of the rule. A cellular automaton can be completely defined by its radius r and a transition rule; moreover, if the latter is linear, the coefficient matrix $\mathbf{A}$ and r are only needed. As an example, Fig. 3 depicts the scheme of a simple 1-D cellular automaton algorithm, made of N cells, radius $r = 1$ and transition rule G. For more details on cellular automata refer to [17, 18].

It is straightforward to identify a connection between FDSs and CA; in fact, several studies have been published on this topic [17]. Some of them have a general focus on the simulation of PDEs [19, 20], and others more particularly on the discretization of waveforms equations [21, 22]. Both FDSs and CA deal with an evolution of state variables on a discrete space-time grid, with the only difference being the fact that cellular automata operate on discrete states, while differential equations look for a continuous domain solution. However, in computer simulations numerical solutions are always discrete, since the processors' resolution is limited by the number of bits. Expressing a FDS as a CA might seem counter intuitive. In fact, the latter are mostly used with a small number of states (sometimes even 2), as they allow us to obtain complex behaviour from very simple rules [23], while in the case of PDEs simulation we deal with an enormous number of possible states. Nevertheless, the CA formalism allows us to obtain a standardized way for expressing FDS in Faust, as it will be shown in the next section.

4.2 The Library

The cellular automata formalism can help us coding standard routing functions for different space dimensions, that properly route the needed neighbours' signals relying only on the information provided by a predefined radius r. Moreover, as it was decided to focus on linear schemes, the transition function for the CA can be simply defined by some coefficient matrices. Contrary to cellular automata, a FDS usually depends on delayed versions of the neighbours' states; therefore, a *time coefficient* t had to be also

taken into account, which indicates how much steps back in time are needed (i.e., if $t = 1$ it means that the maximum delay needed for a neighbour state is 1 sample). With a little abuse of notation, we can express the operation performed by each cell (point) in a D-dimensional scheme as:

$$u_{\mathbf{x}}^{n+1} = \sum [\mathbf{A}_0 \odot \mathbf{N}_r(u_{\mathbf{x}}^n) + \\ + \mathbf{A}_1 \odot \mathbf{N}_r(u_{\mathbf{x}}^{n-1}) + \\ + \cdots + \mathbf{A}_t \odot \mathbf{N}_r(u_{\mathbf{x}}^{n-t}) \,] + F_{in} \tag{13}$$

where $\mathbf{x} = (x_1, \ldots, x_D)$ represents a spatial multi-index, depending on the scheme dimension D, $\mathbf{N}_r(u_{\mathbf{x}}^n)$ is a matrix containing the states of the neighbourhood of $u_{\mathbf{x}}$ at time step n, $\mathbf{A}_i$ are the coefficient matrices for each delayed version of the states, F_{in} is an external signal used to interact with the mesh, the operator $\odot$ represents element-wise matrix multiplication and the non-indexed sum indicates a summation over all the matrix entries. Both $\mathbf{N}_r$ and $\mathbf{A}_i$ contain $(2r+1)^D$ elements. The next paragraphs will detail how this approach can be used to build a FDS model with `fds.lib`.

4.2.1 Defining the Model

In order to code a new model, the user needs to provide: a neighbourhood radius r, a time coefficient t, the number of scheme points (size of the mesh) and the coefficient matrices relative to each point. The latter then need to be ordered in parallel to build a *coefficients scheme*, similarly to what was done in listings 2 and 5. This operation allows us to provide different coefficients for different scheme points, which is essential both for providing boundary conditions other than the clamped ones, and for building physical models with spatially-varying characteristics. As an example, listings 7 and 8 show the definition of the coefficient matrices for equations (5) and (8).

```
r=1; t=1; lambda=c*k/h;
A=2*(1-lambda^2); B=lambda^2; C=-1;
midPoint=B,A,B;
midPointDel=0,C,0;
leftPoint=0,A,2*B;
leftPointDel=0,C,0;
scheme(nPoints) = leftPoint,leftPointDel,
  par(i,nPoints-1,midPoint,midPointDel);
```

Listing 7. 1-D wave equation coefficient matrices

In this code a different coefficient matrix has been set to the leftmost point, in order to apply a Neumann free condition. Since the order of the points goes from left to right, the boundary condition is placed first inside the `scheme` function. Notice that the matrix for the non-delayed states is placed first; this order is very important for the correct functioning of the scheme.

```
r=1; t=1; lambda=c*k/h;
B = lambda^2;
A = 2*(1-2*lambda^2); C = -1;
midPoint = 0,B,0,
           B,A,B,
           0,B,0;
midPointDel = 0,0,0,
              0,C,0,
              0,0,0;
scheme(X,Y) = par (i, X,
  par(j,Y, midPoint,midPointDel));
```

Listing 8. 2-D wave equation coefficient matrices

In this case no boundary conditions have been specified; therefore clamped conditions are implied.

Looking at the code, a visual similarity between the coefficient matrices `midPoint` and `midPointDel` and the stencils in Fig. 1 might be identified. In fact, what we defined here are exactly the coefficients to be applied to the black points in the stencils. In the 2-D case, the CA neighbourhood is necessarily squared; therefore, zeros need to be placed at the corner points in this equation case, as seen in listing 8.

4.2.2 Model Construction

Once the coefficients are defined, the user can simply call `model1D` or `model2D` in order to obtain a fully working physical model. These functions take as inputs the number of points, the radius r, the time coefficient t and the coefficients scheme, and build a model with the same technique detailed in section 3. The built model will have open connections for the forces (or for a generic external signal), one for each scheme point, and will output each point's current state. Interpolation functions can be used in order to correctly select the zone of the mesh to excite or to read the signal from. Fig. 4 depicts the block diagram for a 3 points long version of the scheme in listing 7, built with `model1D`. This diagram will produce the same C++ code as the one represented in Fig. 2.

4.2.3 Interpolation

The library provides linear interpolation operators in 1 and 2 dimensions: `linInterp1D` and `linInterp2D`, which can be used to drive the force to the correct blocks. The index can be a float number varying at run time. These are essentially Faust implementations of the $J(x_i)$ operator, the linear interpolator described by Bilbao [5], not scaled by the spatial step, and work in a similar way as the Faust function `selectoutn` (included in the Faust *basics* library), except that they have the same number of input/output connections; and allow us to use float indexes. The library provides also stairs selectors functions, which only permit to use integer indexes: these are useful in case interpolation is not needed and require less computational power. All these functions are present also in an "out" version that sums all the outcoming signals together, in order to get a mono output signal.

4.2.4 Routing

The functions `route1D` and `route2D` are used to route the forces, the coefficients scheme and the neighbours' signals in the correct places. These are essentially versions of the functions reported in listings 3 and 6, modified to be automatically built starting from the values r and t, and to include also the coefficients matrices in the routing. The routing functions take as input, in this order: the coefficients block, the feedback signals and the forces. In return they provide for each scheme point (in order): the force signal, the coefficient matrices, and the neighbours' signals.

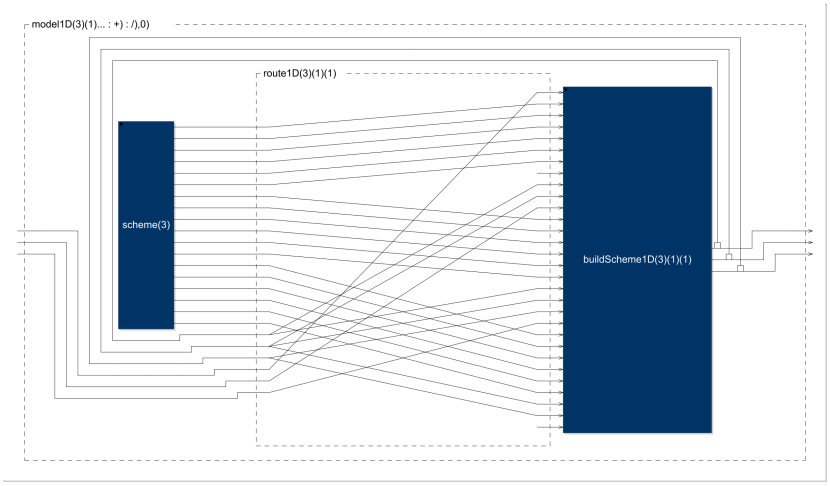


Figure 4. Block diagram for a 3 points long 1-D wave equation scheme, built with the library functions

4.2.5 Scheme Operations

As done previously in section 3, the actual equations are calculated inside some *scheme points blocks* stacked in parallel. The calculation is performed by the `schemePoint` function, which is built based on r, t and D. This function takes as inputs (in order): the force, the coefficient matrices and the neighbours' signals and outputs the next point state according to equation (13). The functions `buildScheme1D` and `buildScheme2D` are used to stack in parallel the proper number of scheme points.

4.2.6 Interaction Models

Even if only linear schemes were considered, nonlinear interaction models can be implemented. The library provides two force functions: a bow and a hammer. The first one is based on the formulation of the Helmholtz motion by Bilbao, the second one is modeled as a dampened oscillator interacting with the mesh in a nonlinear way [5]. Both models need to be coupled with the scheme, in particular, the hammer model involves solving another FDS (the oscillator) in parallel with the mesh. Coupling can be implemented by placing another feedback loop outside the model which drives back the mesh and the oscillator outputs; the interpolation functions can be used to pick the desired output signal from the mesh. Then, the two signals are used to calculate the force, which is subsequently sent as an input to the mesh and the hammer again. The hammer oscillator is integrated inside the hammer function, so that the two force models can be implemented with the same structure. The `bow` block takes as an input (in order): the bow velocity, the force scaling coefficient, the nonlinear parameter and the time step. The `hammer` block takes, in order, the force scaling coefficient, the oscillator frequency ω_0^2, the oscillator damping coefficient σ_0, the hammer stiffness parameter, the nonlinear parameter, the time step and the initial distance between the hammer and the mesh. Both blocks output a force times a scaling coefficient.

5. DISCUSSION & FUTURE WORK

The `fds.lib` library comes with a few examples serving as use-cases (in particular for what concerns the use of the interaction models) and ready-to-use virtual instruments. The modular structure of the library makes it convenient for different purposes. On one hand, the user interested in implementing standard linear schemes can simply write an equation with the desired numerical coefficients and use the model construction functions to easily obtain a working mesh. On the other hand, the library functions can be employed or modified individually to obtain a different result. For instance, the scheme point function could be easily adapted for calculating nonlinear equations, with the only constraint being that the nonlinearities would still need to depend only on the neighbors' states.

The 1-D schemes that were tested performed reasonably well: the CPU load, on an Intel i7-4710HQ processor, was always less than 10% even with a high number of points (schemes with up to 400 points compile and run even on the Faust online IDE). Nevertheless, more systematic performance tests need to be conducted in the future. The 2-D schemes, however, presented an issue. The generated C++ code is completely unrolled, resulting in very long algorithms. As a result, if the number of points is too high, the C++ compiler crashes, an issue similar to the one encountered by Barkati et al. [24]. Our experiments showed that the GCC compiler could not handle meshes with more than 20-by-20 points, which is not enough for many models. This is an issue inherent to the Faust compiler, which needs to be addressed in order to make Faust a complete language for coding FDSs. One way to solve this problem would be to make the compiler able to recognize the parallel structures inside the code and roll them up in the process of translation into C++. However, this topic needs to be more thoroughly investigated. Possible future work includes the implementation of initial conditions for the mesh points and, in the long run, support for implicit schemes.

6. CONCLUSIONS

This work aimed at introducing FDS synthesis in Faust. First, a routine for coding linear explicit physical models in a functional programming way was presented. Then, an overview on the connection between FDS and CA was provided, and the `fds.lib` was introduced: a library that takes advantage of this connection to ease the development FDSs. Together, the two tools allow users interested in implementing these models to exploit Faust potentialities in terms of code translation, optimization and wrapping system. The performances reported by 1-D schemes are promising; however, more methodical tests need to be performed. On the contrary, Faust presented several limitations for compiling 2-D models; these may be addressed by modifying the Faust compiler, making it able to recognize parallel structures.

7. REFERENCES

[1] G. De Poli and D. Rocchesso, "Physically-based sound modelling," *Organised Sound*, vol. 3, pp. 61–76, 04 1998.

[2] J. Smith, "Physical modeling using digital waveguides," *Computer Music Journal*, vol. 16, p. 74, 1992.

[3] C. Cadoz, A. Luciani, and J. L. Florens, "Cordis-anima: A modeling and simulation system for sound and image synthesis: The general formalism," *Computer Music Journal*, vol. 17, no. 1, pp. 19–29, 1993.

[4] J. Adrien, "The missing link: modal synthesis," *Representations of Musical Signals*, 1991.

[5] S. Bilbao, *Numerical Sound Synthesis*. Chichester, UK: John Wiley & Sons, Ltd, 2009.

[6] S. Bilbao, C. Desvages, M. Ducceschi, B. Hamilton, R. Harrison-Harsley, A. Torin, and C. Webb, "Physical modeling, algorithms, and sound synthesis: The ness project," *Computer Music Journal*, vol. 43, no. 2-3, pp. 15–30, 11 2020.

[7] Y. Orlarey, D. Fober, and S. Letz, "FAUST : an Efficient Functional Approach to DSP Programming," in *New Computational Paradigms for Computer Music*, E. D. France, Ed., 01 2009, pp. 65–96.

[8] R. Michon, J. Smith, and Y. Orlarey, "New Signal Processing Libraries for Faust," in *Linux Audio Conference*, Saint-Etienne, France, 05 2017, pp. 83–87.

[9] J. O. Smith, "Virtual electric guitars and effects using faust and octave," in *Proc. 6th Int. Linux Audio Conf. (LAC2008)*, vol. 33, no. 3, Toronto, Canada, 01 2008, p. 1–10.

[10] R. Michon and J. Smith, "Faust-stk: A set of linear and nonlinear physical models for the faust programming language," in *Proceedings of the 14th International Conference on Digital Audio Effects, DAFx 2011*, Paris, France, 01 2011, pp. 199–204.

[11] P. Cook and G. Scavone, "The synthesis toolkit (stk)," in *Proceedings of the ICMC*, Beijing, China, 10 1999.

[12] R. Michon, J. Smith, C. Chafe, G. Wang, and M. Wright, "The Faust Physical Modeling Library: a Modular Playground for the Digital Luthier," in *International Faust Conference*, Mainz, Germany, 07 2018.

[13] E. Berdahl and J. Smith, "Modular and open-source sound synthesis using physical models," in *Proceedings of the Linux Audio Conference*, Stanford, USA, 05 2012.

[14] J. Leonard, J. Villeneuve, R. Michon, and Y. Orlarey, "Formalizing mass-interaction physical modeling in faust," in *Proceedings of the Linux Audio Conference*, Stanford, USA, 03 2019.

[15] C. Erkut and K. Matti, "Digital waveguides versus finite difference structures: Equivalence and mixed modeling," *EURASIP Journal on Advances in Signal Processing*, vol. 2004, pp. 1–12, 06 2004.

[16] C. Erkut and M. Karjalainen, "Finite difference method vs. digital waveguide method in string instrument modeling and synthesis," *Proceedings of the International Symposium on Musical Acoustics (ISMA-02), Mexico City, 2002*, 12 2002.

[17] P. Narbel, "Qualitative and quantitative cellular automata from differential equations," *Lecture Notes in Computer Science*, vol. 4173, pp. 112–121, 10 2006.

[18] X.-S. Yang and Y. Young, *Cellular Automata, PDEs, and Pattern Formation*. Chapman & Hall/CRC, 09 2005, ch. 18, pp. 271–282.

[19] B. Strader, K. Schubert, E. Gomez, J. Curnutt, and P. Boston, "Simulating spatial partial differential equations with cellular automata." in *Proc. International Conference on Bioinformatics & Computational Biology, BIOCOMP*, Las Vegas Nevada, USA, 07 2009, pp. 503–509.

[20] G. Y. Vichniac, "Simulating physics with cellular automata," *Physica D: Nonlinear Phenomena*, vol. 10, no. 1, pp. 96 – 116, 1984.

[21] D. N. Ostrov and R. Rucker, "Continuous-valued cellular automata for nonlinear wave equations," *Complex Systems*, vol. 10, pp. 91–119, 1996.

[22] D. Barkley, "A model for fast computer simulation of waves in excitable media," *Physica D: Nonlinear Phenomena*, vol. 49, no. 1, pp. 61 – 70, 1991.

[23] S. Wolfram, "Cellular automata as models of complexity," *Nature*, vol. 311, pp. 419–424, 1984.

[24] K. Barkati, H. Wang, and P. Jouvelot, "Faustine: A vector faust interpreter test bed for multimedia signal processing," 06 2014, pp. 69–85.

WAVESET TRANSFORMATIONS: SOURCE CHARACTERISTICS AND TRANSFORMATION PECULIARITIES PRODUCING HARDLY PREDICTABLE SOUND RESULTS

Fellipe MARTINS (fmartins@ufmg.br) (0000-0002-2686-6203)[1] and
José Henrique PADOVANI (jhp@ufmg.br) (0000-0002-8919-7393)[1]

[1] *School of Music*, **Federal University of Minas Gerais (UFMG)**, MG Brazil

ABSTRACT

The wavesets are defined as a signal portion between three consecutive zero-crossings and were proposed by Trevor Wishart as a form to manipulate and transform sound files. For simple sounds, wavesets tend to coincide with the wave cycle, but for complex and polyphonic sounds the wavesets can represent a small portion of longer oscillations or retain a slow oscillation superposed by faster ones. As wavesets' shape, duration and amplitude are strongly unique for each sound, modifying a waveform based on this granular criterion can lead to unique sonorities. Firstly, we discuss the main issues that cause highly different sound results when applying the same waveset transformation to different waveforms (e.g., DC, low-frequency content, beats, phase, and others), secondly, we selected several waveset transformations to evaluate the range of sonorities produced as well as contextualizing them in the field of well-known digital audio effects. In the end, we evaluate the major pitfalls of working with this technique and report some enhancements that are currently being studied.

1. INTRODUCTION

Wavesets were proposed by Trevor Wishart [1] as a criterion to execute granular transformations on recorded audio samples. A waveset is defined as the signal portion between three consecutive zero-crossings, which corresponds to a full cycle on a purely sinusoidal oscillation. This definition can be interpreted as an attempt to select grains whose edges tend to zero thus reducing the introduction of high-frequency content due to discontinuities and fast transitions, in a similar fashion to the windowing process used on more traditional granular transformations [2].

The first computational implementation of waveset transformations was released on the Composers Desktop Project (CDP) [3] – a suite of functions for transforming audio files in a non-realtime fashion. As the method requires an analysis stage to find three zero-crossings and several of the transformations deal with the combination of a group of consecutive wavesets, real-time implementations entail a reasonable amount of delay. Real-time implementations available include a VST plugin implementation [4], a Csound operator [5], and several using the SuperCollider environment. The first SuperCollider implementation was made by de Campo [6, 7] as a Quark that analyzes a buffer in non-realtime and then applies transformations in a real-time fashion. An updated version of this Quark was made by de Campo, Bovermann and Rohrhuber [8]. There are also four other implementations Hochherz [9], Nishino [10], Mayer [11] and Seidl [12], the latter working exclusively in real-time.

Waveset transformations are categorized on the CDP as "distortion", a term with broad meaning regarding sound results, ranging from the loss of information (bitcrushing) to the increase of harmonic content through clipping. There is a brief discussion about the wavesets usage, source considerations, sound results, and predictability on [2, 3], but we could not find elsewhere an extensive discussion that aiming the musical usage. The closest related aspects to this topic, which an abundant amount of works can be found, are the evaluation of zero-crossing rate for time series analysis and stochastic processes [13], pitch, noisiness, voice/unvoiced signal detection [14, 15].

2. WAVESETS CONTENT

For stationary harmonic signals, the content of a waveset depends on the number of harmonics, their intensity, and their phase relationship. The first two are well known for defining the perception of timbre, pitch, and beats, however, for signals that have the same number and intensity of harmonics, the phase relationship between them change the wave shape drastically, but it does not interfere on the auditory perception of pitch and timbre, see Fig. 1. As a consequence, when executing waveset transformations on different waveforms that sound the same, it is possible to obtain utterly different graphical and sound results.

As stated by the Fourier transform, a general shape signal can be decomposed as a sum of sinusoidal waves. Therefore we might investigate how the properties of adding two elementary oscillations affect the content of a waveset. As a consequence of a trigonometric identity, the sum of two oscillations is identical to the product of two other oscillations: one proportional to the sum of the frequencies and the other proportional to the difference. Perceptually, the addition of two sine waves can result in four different auditory phenomena: beats, roughness, timbre, and polyphony. In terms of the waveset content, each case will present its

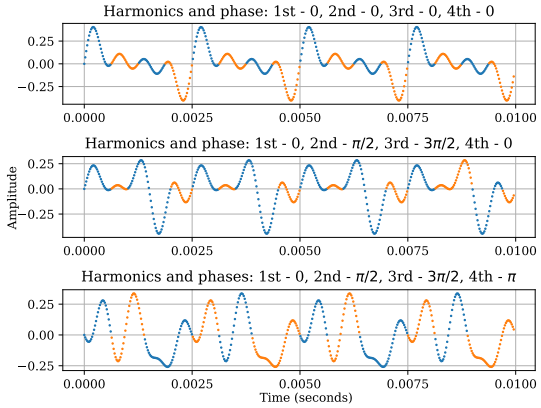


Figure 1. Three tones with four harmonics in the same frequency relationship, same amplitude but different phase relationships.

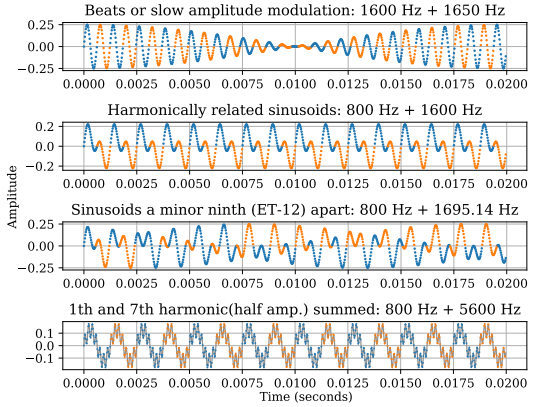


Figure 2. Four auditory effects of sinusoidal addition: beats, harmonically related sinusoids, polyphony and amplitude modulation.

own particularities.

When beats or slow amplitude modulations occur Fig. 2, the content of each waveset tends to retain the carrier shape (i.e. the high-frequency content). However, the slow amplitude variations will change the amplitude of each waveset over time and overmodulation can promote more complex patterns. In the case of harmonically related sinusoids, we will have a constant waveset content within the fundamental period Fig. 2, but there will be faster fluctuations inside this period. As a consequence, the wavesets will lose the characteristic shape of sinusoidal cycles and a full fundamental period will be a compound of more than two wavesets. The third case on this figure shows that when the oscillations of two sinusoids are not harmonically related (e.g. polyphonic signals) the wavesets' content and period will vary over time in a fashion that is strongly dependant on the frequency content of its derivated waves. The last example shows the summation of two highly spaced frequencies, which can be perceived as a single pitch with a complex timbre or as two

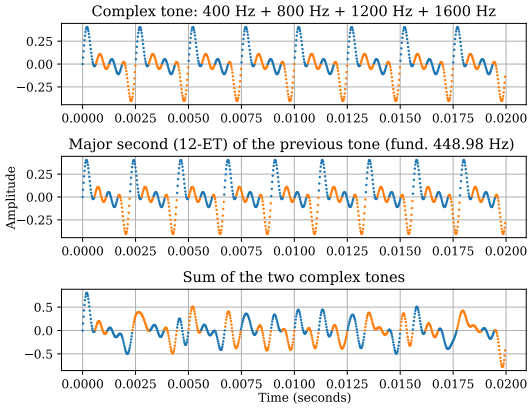


Figure 3. Two complex tones, a 200 cents apart, and the summation of both.

different tones played together. Graphically, we will have a situation of small period wavesets pushed upwards by a slower oscillation.

The last situation described can occur quite often, due to DC offset and non-audible low-frequency content presence in the analyzed signal. In this case, a single waveset may hold a considerable amount of high-frequency content although retaining a long waveset period (e.g. an exaggeratedly long waveset full of high-frequency content displaced upwards due to a very slow frequency component).

One step further regarding complexity, when evaluating the resulting wavesets formed by two spectrally complex tones, separated by a major second and played together, as displayed in Fig. 3, it is possible indeed to recognize its deterministic structure, although one might image several problems related with the transformation of such variable segments. Here, the wavesets' graphical content gets strongly complex and this example, however, is far from the complexity of recorded sounds, illustrating how the transformation of wavesets faces intricate issues. All these temporal variations on the waveset content will increase complexity when basic sound phenomena and properties like energy envelope, reverberation, background noise, etc, are present on the contemplated signal.

3. DISTORT TRANSFORMATIONS

3.1 Overall

As shown in Table 1, the CDP classifies waveset transformations under the umbrella term "distortion" [3]. In general terms, the distort functions available on the CDP either modify the wavesets order (shuffle, repeat, omit) or modify their content (average, clip, etc).

These functions were mainly named "distort" because, in general terms, they introduce high-frequency content and noise to the signal, which is mainly perceived as distortion. Although this nomenclature can be meaningful for sound sources like human-voice and musical instruments, in which radical modifications of the content are perceived by the listener as a degradation of the signal message (or

Function	Description
Average	Average the waveshape over N 'wavecycles'
Cyclecnt	Count 'wavecycles' in soundfile
Delete	Time-contract soundfile by deleting 'wavecycles'
Divide	Distortion by dividing 'wavecycle' frequency
Envel	Impose envelope over each group of cyclecnt 'wavecycles'
Filter	Time-contract a sound by filtering out 'wavecycles'
Fractal	Superimpose miniature copies of source 'wavecycles' onto themselves
Harmonic	Harmonic distortion by superimposing 'harmonics' onto 'wavecycles'
Interact	Time-domain interaction of two sounds
Interpolate	Time-stretch file by repeating 'wavecycles' and interpolating between them
Multiply	Distortion by multiplying 'wavecycle' frequency
Omit	Omit A out of every B 'wavecycles', replacing them with silence
Overload	Clip the signal with noise or a (possibly timevarying) waveform
Pitch	Pitchwarp 'wavecycles' of sound
Pulsed	Impose regular pulsations on a sound
Reform	Modify the shape of 'wavecycles'
Repeat	Timestretch soundfile by repeating 'wavecycles'
Repeat 2	Repeat 'wavecycles' without timestretching
Replace	The strongest 'wavecycle' in a cyclecnt group replaces the others
Replim	Timestretch by repeating 'wavecycles' (below a specified frequency)
Reverse	Cycle-reversal distortion in which the 'wavecycles' are reversed in groups
Shuffle	Distortion by shuffling 'wavecycles'
Telescope	Time-contract sound by telescoping N wavecycles into 1

Table 1. Description of distort functions in the CDP Documentation [3].

distortion), it can be inaccurate or misleading when dealing with electronic sounds and some instruments like guitars, percussion, electric pianos, etc. For this reason, the documentation sometimes uses the term "constructive distortion" for referring to distinct sound results obtained when using these functions. In the following sections, we are going to delineate some of the broad results covered by the category "distortion".

The identification of wavesets is done through zero-crossings counting. However, the zero-crossing rate is also an indication of *pitchness* and *noisiness* of the signal. When the zero-crossing rate is stable at a low value, we have a rough indication that the signal holds a steady pitch characteristic, on the contrary, when the zero-crossing rate is high (being stable or not in this case) it strongly indicates that we have the presence of noise, harshness and signal instability. In this way, a criterion for selecting the minimal waveset length usually helps to avoid operating on the noise content of the sound file and reduces the introduction of harsh and noisy characteristics on the transformed signal.

3.2 Distortion and non-linearity

The term distortion is more commonly used by the audio community as a synonym of harmonic distortion, when a system applies a non-linear [1] curve to its inputs, especially clipping and soft clipping caused by amplifiers, filters, valves, pedals, and other devices, introducing harmonics which were not present in the original signal [16].

Another use for the term *distortion* in the audio community refers to procedures in which information is lost or degrade. Two commonly used types of this effect are sample-rate reduction (samples are discarded thus reducing the bandwidth) and resolution reduction (amplitude quantization bits are discarded).

Besides these two main types, several other digital audio effects operate with strongly non-linear characteristics: compressors, limiters, de-essers, analog simulators (tape, valve, amps, etc), exciters, enhancers, etc. Purely linear conditions are often a too hard constraint, and some distortion is always expected in any kind of audio system.

3.3 Wavesets transformation introducing non-linearities

Operations with wavesets are essentially non-linear procedures because each waveset is going to be processed individually, so each block is subjected to the same wave manipulation, independently of its content. Instead of uniformly transform the signal, sample-by-sample, we are introducing discontinuities due to the block processing. As a frequent result, this may either generate subsequent wavesets with amplitude gaps on their transitions or may produce radically different waveset content varying at a higher rate – which can be perceived as noise.

As a simple test for this second effect, we produced a sinewave with two different amplitude values for each subsequent waveset, therefore each full wave-cycle would be alternating between these two amplitude values. Under low background noise conditions, for a frequency of 440 Hz when the amplitude difference between the two wavesets was greater than around 0.17dB, it is possible to hear the introduction of new harmonic content. This provides a rough parameter about how sensible our auditory system is to modification in such short fragments.

3.4 Other artifacts introduced

One strategy to avoid the previous problem is to operate on a group of wavesets, therefore introducing discontinuities

[1] A system is called linear if it satisfies the property of superposition, expressed in $Ax_1(n) + Bx_2(n) \rightarrow Ay_1(n) + By_2(n)$ in which two given inputs x_1 and x_2 produce the output y_1 and y_2 with its correspondent scalar multiplication.

at a lower rate and reducing the high-frequency content introduction. Furthermore, another option is to execute modifications that are based on the rearrangement of the wavesets, instead of modifying their shape. These strategies partially solve the "distortion" problem but usually introduce other artifacts, especially a mechanical repetition which is perceived as grains of steady pitch or unnatural periodicity. In general, these strategies do not consider the long-term variations of the sound wave (envelope, vibrato, tremolo, *allure*) and tend to break them cyclically, producing repetitions.

Another procedure suggested on the CDP documentation [3] is to "filter" the application of a waveset transformation, which consists of applying transformations only on wavesets of a certain length. As the random and the harmonic part of the recorded signal are generally fully blended, this procedure leads to utterly unpredictable sound results (or from another point of view, leads to results that are strongly source-dependent). In some cases, this process can modify only a portion of a given signal while, in other cases, the resultant sound may consist of sections in which transformed and non-transformed blocks alternate.

Summarizing, the transformation of wavesets – particularly when made without considering other temporal parameters or without applying any smoothing correction – tends to be highly non-linear, introducing several audible artifacts. Moreover, numerous factors modify the wavesets' shape, duration, and rate of recorded sounds (reverberation, DC component, microphone position, intrinsic source characteristics, etc). As a result, the wavesets transformations presented here tend to produce highly unpredictable and source-dependent sound results.

3.5 Traditional approaches

Typical approaches for managing this issue involve some form of overlap and add (OLA) using tapering windows. The most related method is the pitch-synchronous overlap and add (PSOLA), which analyses the pitch of a signal to place the window so that it fits the pitch wavecycle [16]. Therefore, PSOLA is strongly dependent on the quality of its pitch detection algorithm as well as on the degree of pitch stability, noisiness, and inharmonicity. This method is used mainly for pitch shifting and time scale modification, although for more radical transformation, like those proposed by the CDP, it is not commonly used.

4. TRANSFORMATION CASES

4.1 Time Stretch

Waveset-based time-stretching transformations are made by repeating either a single or a group of wavesets. Sonically, the result is far from the traditional harmonic distortion and can lead to peculiar effects. When single wavesets are repeated in a small amount it can range from an introduction of a sound similar to rubber friction. As the repetition rate increases, the static pitch of each waveset emerges. When a group of wavesets is considered, details of the inner structure of the sound file (e.g. like grains, *allures*, small pitch variations) tend to be revealed due to their repetition. It can also produce steady pitches, but as the reproduction rate tends to not be transformed, it is more common to result in repetitions akin to short loops. Further comparisons and audio examples can be found in [17].

4.2 Wavesets alternation (substitution)

When alternating wavesets between two audio files (CDP's *distort reform* function), especially when working with bigger groups of wavesets, it is possible to obtain some unique mixtures between two sounds. This can be used to emulate the phenomena like multiphonics and crosstalk [17]. Moreover, it is a form of achieving multiphonics-alike sounds for instruments that do not easily allow this technique or to produce cross-over sounds made of wavesets extracted from distinct sources. Additionally, this technique produces several amplitude modulation effects which are strongly source-dependent.

4.3 Some effects on vocal sources

The resultant sonorities from the application of wavesets transformation on vocal sources are remarkably varied and cannot be exhausted. Here we inevitably enter the realm of speech perception, which brings a big amount of new topics to the investigation. To point out a few aspects, three CDP functions promote utterly different results when compared with harmonic distortion: *distort pitchwarp*, *distort replace*, and *distort average*. The first introduces a quaver quality to the voice, approximating it to the vocal characteristics of elder speakers. The second increases this approximation by adding a rough/harsh quality. Moreover, it also introduces some features which allude to a speaker with breathing difficulties. The last function is the one that is more similar to harmonic distortion, although it can be more related to the production of a hoarse voice than to vocal overdrive [18].

4.4 *Distort harmonic* and instrumental usage

Some waveset transformations can operate mainly emphasizing qualities of the sound sources, rather than changing or destroying them. The *distort harmonic* function acts like an additive synthesizer, summing the content of a waveset with its correspondent *waveset harmonic* – that is, integer multiples of that waveform. The main sonic result of this procedure is similar to a resonant filter, enhancing and focusing some spectral information as well as promoting sometimes the sensation of pitch shifting. When tested with orchestral instruments, the results could be aurally related to the change of instrument materials (e.g. wood marimba to glass marimba) or to the effect of highlighting amplitude modulations (e.g. more intensity on the vibrato present at the attack time).

5. CONCLUSIONS

Waveset transformations allow artists to explore constructive distortions and numerous other artifacts as creative resources. Furthermore, it also enables new types of combinations and mixtures between different sounds.

In general, few computer music resources provide waveset transformations, therefore there is still plenty of space for new improvements and propositions. The transformation of wavesets, although seems a straightforward task, demands detailed and careful operations to avoid the excessive introduction of high-frequency content and noise. We could not find in our bibliographical review a criterion or a curve-fitting method to reduce noise, harshness, and distortion. As the proposed auditory experiment showed, it is possible to outline some limits, but more complex transformation strategies still need to be tested and discovered.

6. FURTHER DEVELOPMENTS

Currently, we are developing a Python-based library that implements waveset functions and techniques. The main purpose of the library is to provide regular CDP transformations along with new strategies of waveset manipulations. Moreover, we are investigating the usage of different audio descriptors to inform the waveset transformations by means of audio features.

A second strategy, that has shown interesting initial results, is the application of waveset processes in conjunction with other DSP pre-processing and transforms (FFT, DCT, etc) as an intermediate step of analysis/re-synthesis procedures.

7. REFERENCES

[1] T. Wishart, *Audible Design: A Plain and Easy Introduction to Sound Composition*. Orpheus The Pantomime Ltd., 1994. [Online]. Available: http://www.trevorwishart.co.uk/AuD.html

[2] C. Roads, *Microsound*, 1st ed. The MIT Press, 2001.

[3] C. D. P. L. (CDP). (2015) CDP DISTORT Functions. CDP Documentation Home Page. [Online]. Available: http://www.ensemble-software.net/CDPDocs/html/cdistort.htm

[4] M. Norris. (2014) SoundMagic FX - Effect Documentation. [Online]. Available: https://web.archive.org/web/20140113044635/http://www.michaelnorris.info/soundmagic/effects.html

[5] J. ffitch. (2001) Waveset. Orchestra Opcodes and Operators. [Online]. Available: http://www.csounds.com/manual/html/waveset.html

[6] A. de Campo, "Wavesets," SuperCollider Quarks, 2020. [Online]. Available: https://github.com/supercollider-quarks/Wavesets

[7] ——, "Microsound," in *The SuperCollider Book*, 1st ed., ser. The MIT Press, S. Wilson, D. Cottle, and N. Collins, Eds., 2011. [Online]. Available: https://mitpress.mit.edu/books/supercollider-book

[8] A. de Campo, T. Bovermann, and J. Rohrhuber, "WavesetsEvent," musikinformatik, 2020. [Online]. Available: https://github.com/musikinformatik/WavesetsEvent

[9] O. Hochherz, "SPList, a Waveset synthesis library and its usage in the composition "draussen"," 2008, p. 6. [Online]. Available: http://lac.linuxaudio.org/2008/download/papers/19.pdf

[10] H. Nishino, "LC: A Mostly-strongly-timed Prototype-based Computer Music Programming Language that Integrates Objects and Manipulations for Microsound Synthesis," 2014. [Online]. Available: https://scholarbank.nus.edu.sg/handle/10635/78945

[11] D. Mayer, "miSCellaneous_lib," 2020. [Online]. Available: https://github.com/dkmayer/miSCellaneous_lib

[12] F. Seidl, "Granularsynthese mit wavesets für live-anwendungen," 2016. [Online]. Available: https://www2.ak.tu-berlin.de/%7Eakgroup/ak_pub/abschlussarbeiten/2016/Seidl_MasA.pdf

[13] B. Kedem, "Spectral analysis and discrimination by zero-crossings," vol. 74, no. 11, pp. 1477–1493, 1986.

[14] R. Bachu, S. Kopparthi, B. Adapa, and B. Barkana, "Voiced/Unvoiced Decision for Speech Signals Based on Zero-Crossing Rate and Energy," in *Advanced Techniques in Computing Sciences and Software Engineering*, K. Elleithy, Ed. Springer Netherlands, 2010, pp. 279–282.

[15] F. Gouyon, F. Pachet, and O. Delerue, "On the use of zero-crossing rate for an application of classification of percussive sounds," 2000. [Online]. Available: files/publications/dafx00-gouyon.pdf

[16] U. Zolzer, Ed., *DAFX: Digital Audio Effects*, edição: 2nd ed. John Wiley & Sons, 2011. [Online]. Available: http://dafx.de/DAFX_Book_Page_2nd_edition/index.html

[17] F. Martins, "Estudo exploratório de processos de transformação sonora a partir de trevor wishart: reinvenção e tradução para o ambiente supercollider," 2020. [Online]. Available: https://musica.ufmg.br/lapis/?p=1141

[18] A. Endrich. (2016) CDP Tutorial Workshop 1. CDP Tutorial Workshop 1. [Online]. Available: https://www.composersdesktop.com/workshops.html

A WAVE DIGITAL FILTER MODELING LIBRARY FOR THE FAUST PROGRAMMING LANGUAGE

Dirk ROOSENBURG (droosenb@oberlin.edu) [1], **Eli STINE** (estine@oberlin.edu) [1],
Romain MICHON (michon@grame.fr) [2], and **Jatin CHOWDHURY** (jatin@ccrma.stanford.edu) [3]

[1] *TIMARA*, **Oberlin College and Conservatory**, OH USA
[2] **GRAME-CNCM**, Lyon, France
[3] *CCRMA*, **Stanford University**, CA USA

ABSTRACT

In this paper, we present WDmodels, a wave-digital modeling library for the Faust programming language. Recent advancements have made wave-digital models a popular method for simulating analog audio circuits. Despite this, wave-digital modeling techniques have remained challenging to implement for amateurs due to high model complexity. Our library provides a straightforward platform for implementing wave-digital models as real-time digital audio effects.

In this paper, we demonstrate how WDmodels is used to implement wave-digital models containing nonlinear dipoles, such as diodes, and linear R-type adaptors. We describe the library-specific implementation of the connection tree, a data structure commonly used when implementing wave-digital models. We also detail the use of common wave-digital adaptors that have already been implemented in the library. We show how the library may be extended to complex wave-digital models through the implementation of custom adaptors. In order to demonstrate the flexibility of the library, we also present implementations of several audio circuits, including the equalization section of the Pultec EQP-1a program equalizer. Finally, we compare benchmarks from WDmodels and a C++ wave-digital modeling library to demonstrate code efficiency.

1. INTRODUCTION

Faust is a programming language for digital signal processing (DSP) that has grown in popularity in recent years. Its high-level approach to DSP has led to its use by both musicians and experienced DSP programmers [1]. Furthermore, Faust's ability to compile into highly optimized C++ and other low-level coding languages makes it a platform suitable for large, computationally intensive physical models [2].

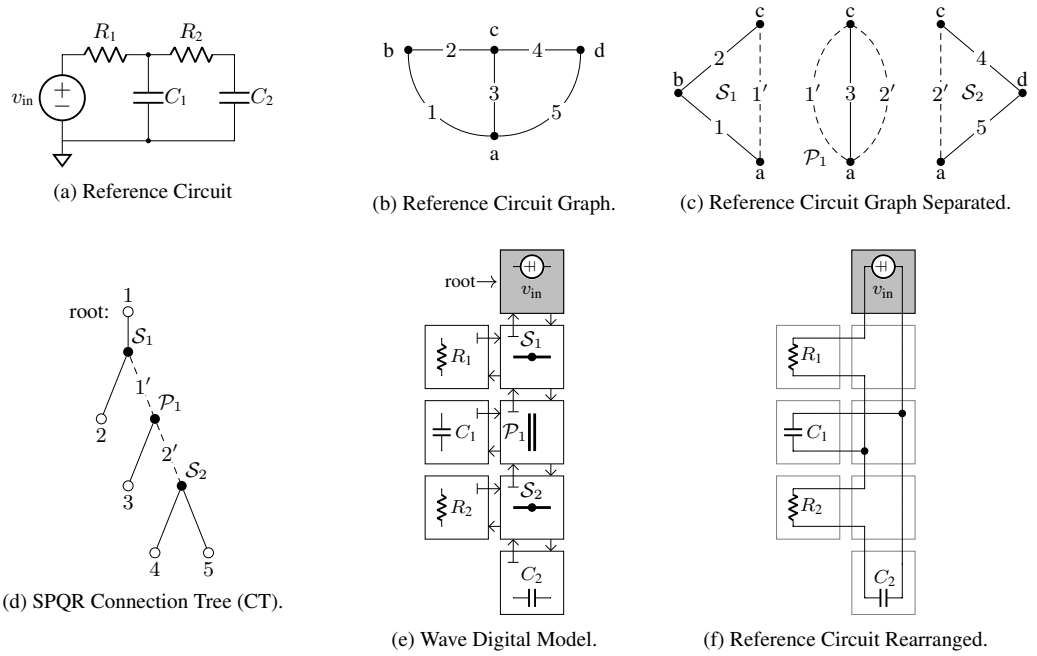
Despite these advantages, Faust's functional method for describing DSP algorithms is incompatible with implementations of physical models that rely on object-oriented data structures. Faust does not currently support direct implementation of multi-directional digital waveguide structures that are commonly found in physical models [3, 4]. To implement a model with multi-directional wave travel in Faust, it must be transformed by inspection into its corresponding direct DSP structure. This process is tedious and unsuitable for large or complex models.

Specific physical modeling methods are supported in Faust through the Faust Libraries [1]. For example, `physmodels.lib` supports the creation of digital-waveguide models of musical instruments by creating custom methods for representing bidirectional traveling waves [5]. `mi.lib` works in conjunction with an external scripting language to generate systems of mass-spring interactions.

In this paper, we present WDmodels, a new addition to the Faust Libraries, that simplifies the creation of wave-digital models of analog audio circuits. Wave-digital models are described by a symbolic representation of the model's connection tree, implemented in Faust using meta-programming. The library uses the Faust compiler to interpret the symbolic connection tree and produce the corresponding direct DSP structure. Many common adaptor types are included in the library, allowing users to easily generate simple models. The library also can be used to create circuit-bendable models of analog-audio circuits for use in real-time processing.

Wave-digital models are a discrete wave-domain representation of physical systems [4]. For circuits, the Kirchoff-based representation is transformed into a traveling-wave-based representation through the bilinear transform. Recent developments extending their capabilities have made wave-digital techniques a popular choice for creating virtual-analog audio effects of analog audio circuits. Their use as a flexible platform for real-time audio simulations has been thoroughly researched [6–8]. Until now, no wave-digital modeling libraries existed in higher-level audio programming languages that are targeted for easy use by artists and musicians. Libraries do exist for C++ [9], a language which is often challenging for beginners, and MATLAB [10], which can be challenging to implement for real-time audio processing. As a result, the modeling technique has remained opaque to many and difficult to learn.

[1] https://github.com/grame-cncm/faustlibraries



(a) Reference Circuit

(b) Reference Circuit Graph.

(c) Reference Circuit Graph Separated.

(d) SPQR Connection Tree (CT).

(e) Wave Digital Model.

(f) Reference Circuit Rearranged.

Figure 1: The process for creating the wave-digital model of a second-order RC lowpass filter using SPQR decomposition.

WDmodels uses a simple process for implementing wave-digital models, allowing those unfamiliar with wave digital modeling to explore the technique for the first time. The block-diagram algebra representation of the DSP processes produced by the library can offer insights into how the compiled model actually functions. The compiled code created by the library runs at speeds comparable to wave-digital model code written in low-level languages. Models may be easily exported to various audio formats for numerous platforms through the `faust2...` tools [11].

In section 2, we provide background information on the creation of connection tree implementations of wave-digital models from analog circuits. Section 3 provides general instructions for using the library. Section 4 introduces two example models and discusses specifics aspects relating to their implementations. Section 5 compares the benchmarking results for WDmodels and a modern wave-digital modeling library written in C++. Section 6 concludes the paper and discusses possible future research directions for this library.

2. BACKGROUND

2.1 Wave-Digital Adaptors

Wave-digital models are networks of connected waveguides called adaptors. When modeling analog circuits, adaptors correspond to parts of the physical system. For example, for each capacitor in the circuit there will be a corresponding adaptor in the wave-digital model. Each adaptor is a wave-scattering junction composed of ports; each port is characterized by an incoming wave, a, a transmitted wave, b, and a port resistance R. The behavior of linear adaptors is described by a scattering equation of the form

$$\mathbf{b} = \mathbf{S}\mathbf{a}, \tag{1}$$

where $\mathbf{b}$ is a vector of transmitted waves, $\mathbf{a}$ is a vector of incident waves, and $\mathbf{S}$ is a scattering matrix. A port of an adaptor is defined to be non-reflective if the port's transmitted wave is not dependent on the current incident wave. An adaptor which includes a non-reflective port is described as "adapted" [4].

The voltage wave definition relates the Kirchhoff behavior of an element to its wave-digital adaptor.

$$\mathbf{b} = \boldsymbol{v} + \mathbf{R}\boldsymbol{i} \tag{2}$$

$$\mathbf{a} = \boldsymbol{v} - \mathbf{R}\boldsymbol{i} \tag{3}$$

Where $\mathbf{v}$ is a vector of voltages across the component, $\mathbf{i}$ is a vector of currents through the component, and $\mathbf{R}$ is a vector of port resistances [12].

Typically, simple circuit components have corresponding one-port adaptors while circuit topology is represented by adaptors with two or more ports. Many linear circuit elements, such as voltage sources, resistors, parallel connections, and series connections, may be digitized directly using the parametric wave definition. Reactive circuit elements are digitized using a conformal frequency mapping, generally the bi-linear transform [8]. The resulting adaptor formulation will rely on sample delay and is said to contain part of state of the system [4].

2.2 The Connection Tree

The connection tree of a model is formed by performing SPQR decomposition on the graph of a circuit. In this process, the circuit's graph is broken into simpler sections by recursively removing series and parallel elements. From

```
1  secondorder(R2, C2, in1) = wd.buildtree(tree)
2  with{
3    //declare components
4    vs1(i) = wd.u_voltage(i, in1);
5    r1(i) = wd.resistor(i, 4700);
6    c1(i) = wd.capacitor(i, 2.2e-6);
7    r2(i) = wd.resistor(i, R2);
8    c2(i) = wd.capacitor_Vout(i, C2);
9    //form connection tree
10   tree = vs1 : wd.series : (r1, (wd.parallel :
       (c1, (wd.series : (r2, c2)))));
11 };
```

Figure 2: The implementation of a second-order RC lowpass filter simulation in Faust using WDmodels. The corresponding wave-digital model is shown in Figure 1

these sections, a tree is formed. Each node in the tree corresponds to an adaptor in the wave-digital model [12, 13]. This process is shown in Figure 1. The leaf nodes (terminating nodes with no downward-going connections) of the connection tree represent circuit components, such as resistors and capacitors. Connection nodes (nodes with both upward-going and downward-going connections) represent circuit topology, denoting elements connected in series or parallel.

Since wave-digital models are a complex network of interconnected scattering junctions, it is critical that adaptors in wave-digital models are arranged in order to prevent delay-free loops within the structure. This process is performed by "adapting" the model, where the port resistances of adaptors are set in order to eliminate these loops by making ports non-reflective. Commonly, this is performed by exploiting the properties of the connection tree. By setting port resistances such that the upward-facing port of each node is non-reflective, this guarantees the resulting structure will contain no delay-free loops. Since the root node is the only node with no upward-facing ports, it is the only node left unadapted. All other nodes within the tree are adapted [14]. Many wave-digital adaptors cannot be adapted, such as ideal voltages sources or nonlinear devices. Thus, an unadaptable adaptor often is chosen as the root of the connection tree.

3. LIBRARY IMPLEMENTATION OVERVIEW

3.1 Approaching Wave-Digital Models in Faust

The library uses meta-programming to simplify implementing a wave-digital model in Faust. For each node in the model's connection tree, a separate function in Faust is declared that describes that node's behavior. To describe the connection tree, the node functions are combined into a single symbolic function using Faust compositional operators. Finally, the symbolic connection tree function is passed to a build-function that generates the model by recursively inspecting the symbolic function.

3.2 Component Declaration

3.2.1 Common Nodes

The library includes many pre-written nodes that correspond to common wave-digital adaptors. These include both nodes of component adaptors, such as resistor, capacitor, and inductor adaptors, and topological adaptors, such as series and parallel adaptors [8, 10]. A current list of all nodes included in the library can be found as part of the Faust Library Documentation [2].

Code lines 4-8, in Figure 2, shows the declaration of nodes for the component adaptors of the second-order RC lowpass filter; its wave-digital model is shown in Figure 1e. Line 5 declares a resistor adaptor node with a component value of $4.7\,\mathrm{k}\Omega$.

```
1  r1(i) = wd.resistor(i, 4700);
```

Note the parameter i, an index parameter required by the model-building function. Each node in the model must be declared using this form. The prefix `u_` to a node name, as seen in line 4, denotes a node corresponding to the unadapted version of that adaptor.

3.2.2 Model Inputs

Model inputs within the library must be declared explicitly as named parameters of the model. A wave-digital model may receive inputs in the form of voltage, resistance, or any other component value. By declaring inputs as parameters of the enclosing function, they may be called explicitly as component values. This convention is shown in Figure 2; the variables R2, C2, and in1 are model parameters that are used to set component values. Note that the empty signal operator "_" should never be used as a component value, as it will break the internal signal flow of the model.

3.2.3 Model Outputs

Outputs of the model are declared by calling specialized nodes, such as in code line 8, Figure 2.

```
1  c2(i) = wd.capacitor_Vout(i, C2);
```

The model will output the voltage across C_2 as an audio signal.

Nodes which include a model output will have a suffix that describes its output. Only voltage across a component (suffix `_Vout`) and current through a component (suffix `_Iout`) are currently supported as possible outputs from the model.

3.2.4 Custom Nodes

Since a wave-digital model might include a specialized adaptor unique to the circuit, the library also includes several functions that help generate nodes from wave-scattering junctions of custom adaptors. Wave scattering junctions can be formed formed according to methods in [3]. Each sequential input-output pair of the scattering junction will correspond to a port of the node. The automatic adapting process requires the scattering junction have the upward-facing port resistances declared as parameters.

`u_genericNode` forms an unadapted node from the scattering junction. `genericNode` forms an adapted node from the scattering junction. The function assumes that the first input-output pair is the non-reflective

[2] `https://faustlibraries.grame.fr/libs/wdmodels/`

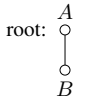
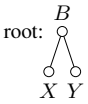
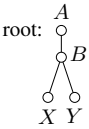
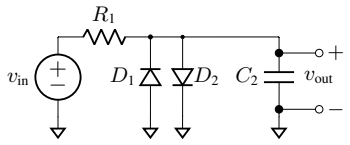
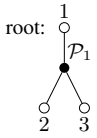
Connection Tree Structure	Representation in Faust
	`1 A : B`
(a) Simplest two-level tree	
	`1 B : (X, Y)`
(b) Layers with multiple elements	
	`1 A : (B : (X , Y))`
(c) Representation of multi-level trees through subtrees	

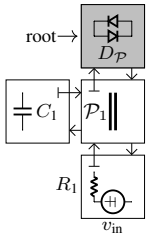
Figure 3: The meta-programming based representation of the connection tree implemented by the library. The sequence composition denotes a downward going connection. Parallel composition denotes a multi-element layer. Sub-tree based associativity is used to represent multi-level trees.



(a) schematic



(b) connection tree



(c) wave-digital model

```
1 diode-clipper(in1) = wd.builtree(tree)
2 with{
3   //declare components
4   c1(i) = wd.capacitor(i, c1)
5   vres(i) = wd.resVoltage(i, in1);
6   dp(i) = wd.u_diodeAntiparallel(i, Is, Vt);
7   //declare connection tree
8   tree = dp : wd.parallel : (c1, vres);
9 }
```

(d) Faust implementation

Figure 4: The wave-digital model and simplified Faust implementation of a one-capacitor diode clipper. The antiparallel diode is modeled using Schottky's diode law and implemented with an iterative Lambert $\mathcal{W}$ function solver.

pair. It also depends on the port resistance rule that creates the non-reflective behavior. `genericNode_Vout` and `genericNode_Iout` form leaf nodes similarly to `genericNode` while also treating the node as a model output.

3.3 Connection Tree Formation

To describe the connection tree, WDmodels implements a custom symbolic representation of trees using existing Faust operators. Sequence composition declares a parent (P) to child (C) relationship between nodes:

$$P : C. \tag{4}$$

If a parent node has multiple children, they are declared in a list using parallel composition:

$$P : (C_1, C_2, \ldots C_n) \tag{5}$$

More complex trees are implemented by using this definition recursively. A complex tree can be broken into simple functions representing subtrees, then each subtree function is treated like a single node. This symbolic representation is detailed for several example trees in Figure 3. The declaration of the connection tree for the model in Figure 2 occurs in line 10.

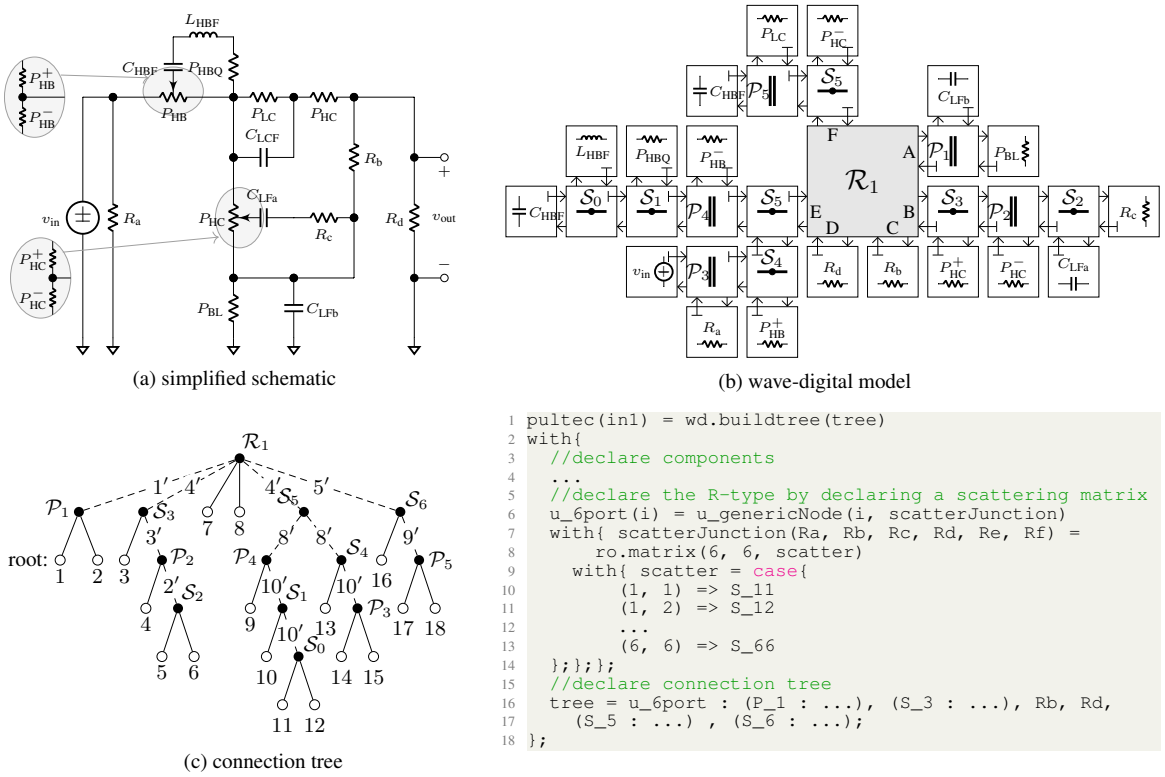
The implemented connection tree must be properly formed along wave-digital model conventions. All nodes in the connection tree must be adapted, except for the root, which must be unadapted. Each node also expects a specific number of parent and child nodes based on its internal characteristics. For example, an adapted node implementing a three-port parallel adaptor must have two children and one parent in the connection tree.

3.4 Building the Model

To create a working model, the connection tree function is passed to the model-building function `buildtree`. This step is declared in Figure 2, line 1. `buildtree` interprets the meta-programming of the connection tree and produces the final model function. As part of the model-building process, the wave-digital model is adapted. The port resistances of all adaptors are automatically set to the proper values.

Most implementations of wave-digital filters rely on a tree data structure to implement the connection tree. Adaptors are implemented as node objects in the tree. A recursive tree traversal of the connection tree data-structure is performed for each computation cycle of the connection tree [9, 13].

The library uses an alternative implementation of wave-digital models. Instead of performing a tree-traversal at each step, we perform a tree inspection during compilation, generating an instructional method for computing the model. In the tree inspection, the `buildtree` recursively inspects the connection tree function and creates three functions corresponding to the computations of downward-going waves, upward-going waves, and root-




```
pultec(in1) = wd.buildtree(tree)
with{
  //declare components
  ...
  //declare the R-type by declaring a scattering matrix
  u_6port(i) = u_genericNode(i, scatterJunction)
  with{ scatterJunction(Ra, Rb, Rc, Rd, Re, Rf) =
      ro.matrix(6, 6, scatter)
    with{ scatter = case{
        (1, 1) => S_11
        (1, 2) => S_12
        ...
        (6, 6) => S_66
  };};};
  //declare connection tree
  tree = u_6port : (P_1 : ...), (S_3 : ...), Rb, Rd,
    (S_5 : ...) , (S_6 : ...);
};
```

Figure 5: The wave-digital model and simplified Faust implementation of the Pultec EQP 1-A passive equalization section. The R-node is implemented by creating 6×6 scattering junction and using library tools to form a custom node.

reflected waves. Each function consists of parametrized node functions and specialized routing for signals.

4. IMPLEMENTATION EXAMPLES

4.1 Diode Clipper

The one-capacitor diode clipper circuit has been thoroughly studied as a virtual analog model [15, 16]. As a wave-digital model, it can be simulated using four adaptors. Figure 4 shows the implementation of a one capacitor diode-clipping circuit as a wave-digital model using WD-models. The resistive voltage source adaptor encloses both v_{in} and R_1. The antiparallel diode pair is modeled using a single nonlinear adaptor chosen as the root of the tree.

The diode pair adaptor is formed using Schottky's ideal diode law, as shown in [17]. The diode adaptor's behavior is described by

$$b = a + 2\text{sgn}(a)\left[RI_s - V_T\mathcal{W}\left(\frac{RI_s}{V_T}e^{\frac{\text{sgn}(a)a+RI_s}{V_T}}\right)\right],$$

where $\mathcal{W}(x)$ is the Lambert $\mathcal{W}$ function, I_s is the diode's saturation current, and V_T is the diode's thermal voltage. The library node which implements an antiparallel diode pair adaptor, `u_diodeAntiparallel`, relies on `lambert`, a custom Lambert $\mathcal{W}$ function approximation that uses Newton-Raphson iteration to approximate the solution [18]. Since compiled Faust code cannot contain loops, `lambert` uses a set number of iterations that will be performed at each sample.

4.2 Pultec EQP-1A

The library also provides several functions which allow the implementation of user-generated adaptors as nodes. This allows for the simulation of complex circuits through the implementation of R-type adaptors.

Here we present an implementation of the Pultec EQP-1a's passive equalization section. The EQP-1a is a program equalizer popular with audio engineers for master bus equalization and general mastering [19]. The equalization is performed by an passive RLC network. The output is then passed through a tube makeup-gain stage.

We implemented the passive RLC network as a wave-digital model as shown in Figure 5. Figure 5a shows the simplified schematic of the EQP 1-A RLC equalization network. By performing SPQR decomposition, we found the connection tree associated with the circuit, shown in Figure 5c. The circuits equivalent wave-digital model is shown in Figure 5b. A resistive voltage source with negligible series voltage was chosen to model the voltage input.

The scattering matrix of this R-type adaptor was derived using methods described in [12]. To implement this adaptor as a node in the library, we used `u_genericNode`. First the adaptor's scattering matrix was used to form a

	C++ WDF		WDModels	
	time (s)	ratio	time (s)	ratio
Second-Order	0.197	50	0.0746	130
Large Network	0.606	16	0.162	61
Diode Clipper	0.176	57	0.507	20

Figure 6: A comparison of computation benchmarks for WDmodels and a C++ wave-digital modeling library. Time to compute for each was averaged over three runs. The ratio of real time to computation time is also displayed for comparison purposes; higher is better.

6×6 scattering junction, `scatterJunction`, using methods in [3]. The six upward-facing port resistances $(R_a, R_b, \ldots, R_f)$ were declared as parameters. The scattering junction was passed to `u_genericNode` to form it into an unadapted node.

5. COMPARISON

To determine both the realizability and optimization of the WDmodels, we benchmarked it against a personal wave-digital modeling library [3] written in C++. The benchmarks were performed on a desktop PC with an AMD Ryzen 2600x processor and 16GB of RAM running Manjaro Linux. Three models were tested: the second-order RC lowpass filter shown in Figure 1e, the diode clipper shown in Figure 4c, and an arbitrary large linear model. The three models were first implemented in both libraries. For the Faust library, the Faust code was then compiled into C++. Each implementation was then tested to determine the time it took to process $10\,\mathrm{sec}$ of randomized audio at $192\,\mathrm{kHz}$ sample rate. A high sample rate was chosen to show the potential for oversampling, as oversampling is commonly used to improve the accuracy of physical models. Three runs for each implementation were performed and times were averaged together to determine a mean computation time for each implementation. The results are displayed in Figure 6. The full code used for testing is available on GitHub [4].

Our benchmarks show that both libraries produce simulations capable of easily running in real time at high sample rates. For the linear networks, WDmodels outperforms the C++ library by a factor of 2-3. This can be attributed to the Faust compiler, which is designed to produce highly optimized DSP processes. It should be noted that the C++ library used is already optimized through templating; comparison to C++ libraries that perform recursion through the model's connection tree in real time would likely have slower performance.

WDmodels is outperformed by the C++ library on the diode clipper by a factor of about 3. This is likely due the C++ library using a more optimized method for computing the Lambert $\mathcal{W}$ function. Further optimization `lambert` would be helpful to improve the performance of the diode clipper.

[3] `https://github.com/Chowdhury-DSP/chowdsp_utils`

[4] `https://github.com/jatinchowdhury18/wdf-bakeoff`

These benchmarks show that WDmodels is an excellent platform for real-time simulation using wave-digital models. The highly optimized code produced by the Faust compiler is suitable for use in digital audio effects or other real-time applications.

6. CONCLUSION

In this paper, we presented a new Faust library, WDmodels. The library greatly simplifies the process of implementing wave-digital models in Faust by creating a symbolic representation of the connection tree data structure using meta-programming. We explained how the symbolic representation is used within the context of the library and showed examples that demonstrate its use.

Typical wave-digital modeling libraries rely on recursion through a tree for the computation of each sample, which can be computationally expensive. In the library, a tree traversal is only performed once at compilation instead of every sample at runtime, resulting in a significant reduction of computational complexity of the model. We have also shown that the Faust compiler is able to produce C++ code that rivals or outperforms C++ wave-digital modeling libraries, in some cases. The code produced is suitable for implementation in real-time digital audio effects.

Currently, the library only includes nodes for some one-port nonlinearties, specifically the diode. Most complex audio circuits rely on complex nonlinear elements, such as transistors and vacuum tubes, which are modeled using multiport nonlinearties [20]. Thus, the development of methods to support multiport nonlinear adaptors in faust would greatly widen the scope of models which can created with default library nodes and functions.

Acknowledgments

Thanks to Kurt Werner for assisting with wave-digital model figures and to Rob Owen, Julius O. Smith, Stéphane Letz, and Yann Oraley for aiding with the development of the library.

7. REFERENCES

[1] R. Michon and Y. Orlarey, "The faust online compiler: a web-based ide for the faust programming language," in *Linux Audio Conference*, 2012. [Online]. Available: http://lac.linuxaudio.org/2012/papers/22.pdf

[2] Y. Orlarey, D. Fober, and S. Letz, "Faust: An efficient functional approach to dsp programming," 2009, unpublished. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02159014

[3] J. O. Smith III, *Audio Signal Processing in Faust*, CCRMA, Stanford, CA, USA, 2020. [Online]. Available: https://ccrma.stanford.edu/~jos/aspf

[4] ——, *Physical Audio Signal Processing*, 1st ed. http://www.w3k.org/books/: W3K Publishing, 2010.

[5] R. Michon, J. Smith, C. Chafe, G. Wang, and M. Wright, "The faust physical modeling library: a

modular playground for the digital luthier," in *International Faust Conference*, 2018.

[6] J. Zhang and J. O. Smith III, "Real-time wave digital simulation of cascaded vacuum tube amplifiers using modified blockwise method," in *Proc. 21th Intl. Conf. Digital Audio Effects (DAFx-18)*, 2018.

[7] J. Chowdhury, "A comparison of virtual analog modelling techniques for desktop and embedded implementations," 2020. [Online]. Available: https://arxiv.org/pdf/2009.02833.pdf

[8] K. J. Werner, "Virtual analog modeling of audio circuitry using wave digital filters," Ph.D. dissertation, Stanford University, 2016.

[9] M. Rest, R. W. Dunkel, K. J. Werner, and J. O. Smith III, "RTWDF—a modular wave digital filter library with support for arbitrary topologies and multiple nonlinearities," in *Proc. 19th Intl. Conf. Digital Audio Effects (DAFx-16)*, 2016.

[10] U. Zölzer, X. Amatriain, D. Arfib, J. Bonada, G. De Poli, P. Dutilleux, G. Evangelista, F. Keiler, A. Loscos, D. Rocchesso *et al.*, *DAFX-Digital Audio Effects*, 2nd ed. John Wiley & Sons, 2011.

[11] D. Fober, Y. Orlarey, and S. Letz, "Faust architectures design and osc support." in *Proc. 14th Intl. Conf. Digital Audio Effects (DAFx-11)*, 2011, pp. 231–236.

[12] K. Werner, A. Bernardini, J. Smith, and A. Sarti, "Modeling circuits with arbitrary topologies and active linear multiports using wave digital filters," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. PP, pp. 1–14, 06 2018.

[13] D. Franken, J. Ochs, and K. Ochs, "Generation of wave digital structures for networks containing multiport elements," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 3, pp. 586–596, 2005.

[14] A. Sarti and G. De Sanctis, "Systematic methods for the implementation of nonlinear wave-digital structures," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 2, pp. 460–472, 2008.

[15] J. Parker, F. Esqueda, and A. Bergner, "Modelling of nonlinear state-space systems using a deep neural network," in *Proc. 23rd Intl. Conf. Digital Audio Effects (DAFx-19)*, 2019.

[16] D. T. Yeh, J. S. Abel, and J. O. Smith, "Automated physical modeling of nonlinear audio circuits for real-time audio effects—part i: Theoretical development," *IEEE transactions on audio, speech, and language processing*, vol. 18, no. 4, pp. 728–737, 2009.

[17] K. J. Werner, V. Nangia, A. Bernardini, J. O. Smith III, and A. Sarti, "An improved and generalized diode clipper model for wave digital filters," in *Audio Engineering Society Convention 139*. Audio Engineering Society, 2015.

[18] S. D'Angelo, L. Gabrielli, and L. Turchet, "Fast approximation of the lambert w function for virtual analog modelling," *Proc. 23rd Intl. Conf. Digital Audio Effects (DAFx-19)*, vol. 100, 2019.

[19] *Pultec model EQP-1A Manual*, Pulse Techniques, Inc., West Englewood, NJ, 1951.

[20] K. J. Werner, V. Nangia, J. O. Smith, and J. S. Abel, "A general and explicit formulation for wave digital filters with multiple/multiport nonlinearities and complicated topologies," in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2015, pp. 1–5.

THE INFORMATICS PHILHARMONIC IN NEW MUSIC

Kaitlin PET (kpet@iu.edu) [1] and **Christopher RAPHAEL** (craphael@indiana.edu) [1]

[1] *Luddy School of Informatics Computing and Engineering*, **Indiana University Bloomington**, IN USA

ABSTRACT

This paper describes our ideas and experience using a musical accompaniment system — the Informatics Philharmonic (Info Phil) software application — in new compositions that combine live performance on an acoustic instrument with computer-generated sound.

Using this system, we adapted three compositions of Jacob ter Veldhuis ("JacobTV") for solo instrument and tape: *Billie*, *Garden of Love*, and *Farewell Feathered Friends*. We will discuss our experience adapting these works, and analyze the adapted versions in terms of enabling the longevity of the composition, musical effect, and performance ease. Lastly, we show how the Informatics Philharmonic technology can be extended to include other programs, such as Max/MSP, to allow virtually unlimited interaction between a soloist and recorded or live-processed sound.

1. INTRODUCTION

Since the 1950s, many composers have chosen to combine acoustic and electronically-generated musical material. Early examples such as Stockhausen's *Gesang Der Junglinge* utilized a completely fixed format, meshing recorded vocal lines with processing and generated sounds [1]. For many decades composers have also incorporated live performers with pre-assembled tape tracks. For example, Steve Reich's *Different Trains* joins a live string quartet with tape composed from processed vocal material and train sounds, and Thea Musgrave's *Niobe* pairs solo oboe with tape containing high vocal-like lines and bells. Unlike completely fixed works, these *mixed* works can vary from performance to performance based on the live player's expressive input. The performer is still entirely responsible for coordinating with the tape, but such pieces need little technological infrastructure; usually only a microphone, playback device and speaker are required for a successful concert.

Another popular mixed composition paradigm combines live acoustic performance with *interactive* electronics. Works such as John Chowning's *Voices*, Pierre Boulez's *Antemes* and George Lewis's *Emergent* pair a live acoustic soloist with electronics that distort and react to their playing in real time. This type of composition allows direct interaction between players and electronics, but often requires nontrivial technological setup and execution. Unless the composer is present at a performance or the performer is especially technology-literate, these complex setups can serve as a barrier to a work's accessibility [2].

The development and maturation of real-time score following technology has brought exciting possibilities to the world of mixed composition. Introduced by Dannenberg and Vercoe in the mid 1980s [3, 4], score following allows a computer to know a player's position in a predetermined score at all times. New music compositions which incorporate score following take different approaches. For example, *Tensio* by Phillip Manoury has a "live media" approach, using the IRCAM Antescofo score following system to automatically trigger electronic events and real-time audio processing in intricate response to a live string quartet [5, 6].

Another approach, advocated by the authors, builds on top of score following by adding a model for musical timing. In the Informatics Philharmonic (Info Phil) software [7], a musical timing model translates the onset times of each soloist note into a continually-evolving prediction of future musical evolution. This model allows the system to coordinate through prediction, rather than response, representing the most important difference between our work and that of IRCAM. These predictions can be used to drive a MIDI accompaniment or to continually modify the playback speed of a pre-recorded accompaniment track. In this context performers are free to choose their tempo at all times, closely simulating the experience of playing with human collaborators.

Examples of this approach in new music include Nick Collin's *Concerto for Accompaniment* for oboe and computer-controlled piano [8]. Collins wrote this piece explicitly for the Info Phil, composing a piano part that was not "humanly performable" then using the Info Phil to synchronize MIDI-generated piano with a live player. Jan Beran also wrote a series of works for Info Phil utilizing a similar approach [9]. A hallmark of these pieces is the use of the unlimited technical virtuosity of the computer, including unplayably fast passages and highly complex rhythms such as simultaneous groups of 7 and 11.

In Fall 2019, we approached Dutch composer Jacob ter Veldhuis — "JacobTV" — about adapting his work *Garden of Love* (2002) for oboe (or soprano saxophone) and tape to work with the Info Phil. He and a longtime collaborator, saxophonist Connie Frigo, suggested we also adapt *Billie* (2003), a jazz-based piece commissioned by Frigo for alto saxophone and tape. Frigo premiered the Info Phil adaption of *Billie* at the 2020 North American Saxophone

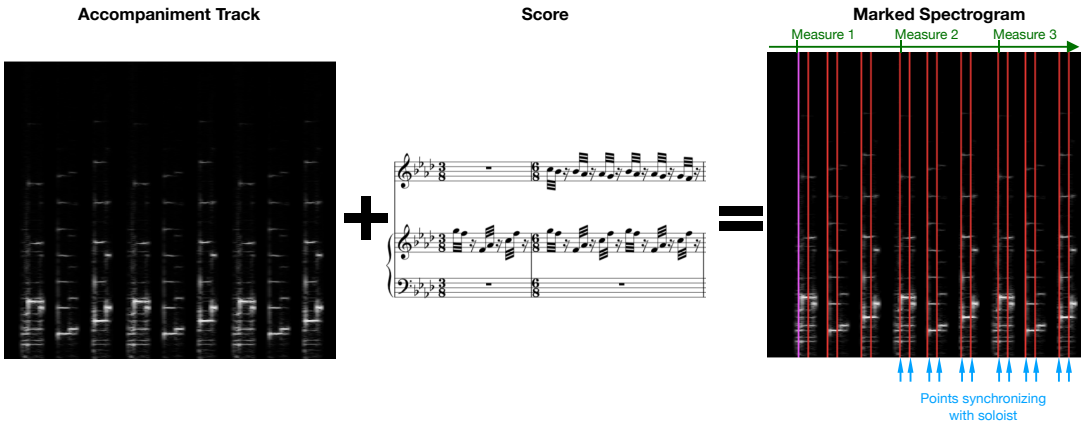


Figure 1: Given a spectrogram of the accompaniment tape and a full score, one can use Info Phil to create a marked track with accompaniment entrances, soloist entrances, and other temporal information.

Alliance (NASA) Biennial Conference. In Summer 2020, we adapted *Farewell Feathered Friends*, a piece composed by JacobTV during the COVID-19 pandemic for piccolo and tape. Our version was premiered at the recital of Jamey Guzman, a master's student at the Jacobs School of Music at Indiana University, Bloomington (JSoM).

2. PROCESS

2.1 Informatics Philharmonic Overview

We will start by discussing the considerations and capabilities of the Info Phil score-following and AI accompaniment software [7]. While the Info Phil can produce MIDI-generated accompaniment, the most common use case involves a prerecorded accompaniment track. Using audio time-stretching techniques, the track is resynthesized in real time to follow a soloist.

2.1.1 Requirements and Operation

The Info Phil requires two items to create its reactive accompaniment: a recording of the accompaniment alone and a full score in symbolic notation. We first generate an "index" into the accompaniment audio by performing offline polyphonic score alignment, followed by some degree of manual adjustment. When this phase is complete, each note in the score is matched to its corresponding time point in the accompaniment recording. See Figure 1 for a visual representation of this process. After this marked accompaniment information is loaded into the Info Phil application, a player simply needs to press the "play" button on the Info Phil's GUI interface to start reactive accompaniment playback.

2.1.2 Score Following and Soloist Prediction Model

During rehearsal or performance, the Info Phil uses a Hidden Markov Model(HMM)-based online score follower to analyze the soloist's movement through the score in real time. In this method, the soloist's audio is read into the Info Phil as 30 milisecond frames. The HMM model relates the sequence of "heard" frames to the sequence of notes making up the monophonic soloist part, allowing the Info Phil to continually determine the performer's position in the marked score. An additional layer — the timing model — uses this information to predict when the soloist's next notes will occur. The track's playback rate is then sped up or slowed down through phase vocoding to maintain the correct rhythmic relationship between solo and accompaniment.

A high-level example showing how the Info Phil's audio recognition and prediction systems work together in real time can be seen at this link: `http://www.music.informatics.indiana.edu/~craphael/info_phil/info_phil_2012/Maria_RC.mp4`, which shows an Info Phil-based performance of the 4th movement of Lalo's *Symphonie Espagnole*.

2.1.3 Training

The Info Phil will also learn from a player's past performances, should they choose to "train" on that data. Information about the player's past tempo tendencies allows the Info Phil to better anticipate their future interpretation by estimating free parameters of the timing model.

2.1.4 Audio Stretching and Compression

The Info Phil utilizes a generic variety of phase vocoding which does not distinguish between pitch and noise-based content. This means that when the soloist plays at a different tempo from the original accompaniment recording, the speedup or slowdown in accompaniment playback rate is the same regardless of what content is being stretched or compressed. A side effect of this phase vocoding strategy is that noise-based sounds which should take the same amount of time regardless of tempo (e.g. percussion, wind entrances, and non-traditional music elements like found sounds and birdsong) are altered to be faster or slower than their normal length. Usually this change is not noticeable to the human ear, but may sometimes sound unnatural when extreme playback rate changes are applied.

We found that training the system on previous performances was generally effective in decreasing these arti-

facts. Usually, the most extreme shifts occur when the Info Phil encounters a player's note in an unexpected location, then must quickly move through the audio track to re-align. Training allows the Info Phil to learn the player's tendencies, and thus helps protect against extreme track modifications likely to result in artifacts.

2.2 Adapting Fixed Media Works

In adapting JacobTV's music, the track marking process described in Figure 1 is complicated by his extensive use of hard-to-notate sound sources like spoken voice and birdsong. The Info Phil was constructed for traditional Western art music, thus assuming discrete pitches and "rational" rhythm. Birdsong and spoken voice have many microtonal shifts and slides, and thus are hard to notate within a traditional written music framework.

Fortunately, JacobTV's score already converted these sounds to notes and rhythms in order to help the soloist better interpret their rhythmic relationship with the accompaniment track. Since this notation was meant as a performer's guide rather than an absolute specification, the track labeling process in Figure 1 was only semi-automatic, relying heavily on manual input.

This indirect way of translating voice and birdsong into absolute rhythms also has another effect: the "true" rhythm specification is a compromise between the written score and the tape track. In JacobTV's music, live players often need to play in unison with birdsong or voice. In the preface to *Billie*'s score, Connie Frigo explains that in these cases, if a player finds slight deviations between the notated rhythm and Billie Holiday's voice samples, they should always assume the audio recording is correct and alter their rhythm to match vocal inflections. This performance practice poses a potential issue for Info Phil adaptation. Since the program bases its behavior on the score, it only seeks simultaneity between a solo event and an accompaniment event if both events are explicitly notated at the same score position. To combat this issue, we edited the score so true simultaneous sections always had equivalent rhythmic notation. As long as the birdsong or voice "notes" occur at the same rhythmic positions as soloist notes, the Info Phil tries to align these events during performance.

3. JACOBTV'S PIECES

3.1 Garden of Love

Our first adaptation was *Garden of Love*, originally for oboe and tape, though now more well-known as a piece for soprano saxophone. Based on the poem "Garden of Love" by William Blake, the piece uses spoken audio samples to create rhythmic "voice melodies." JacobTV combines these vocal samples with synthesized sixteenth notes and birdsong to create a driving, high-energy, at times mechanical piece. While rehearsing *Garden of Love* we observed that the Info Phil facilitated the performer's task by allowing pauses to enable breathing and otherwise assisting coordination. Since the original version requires almost continual playing to keep up with the tape, we found that slight breaks between sections made playing the piece a more enjoyable experience. We also noticed that the Info Phil was able to automatically get back on track when the soloist made rhythmic or timing errors.

This sense of performance ease was reinforced by Wes Taylor, a JSoM saxophone student whom we worked with. He appreciated that similar to performing with a live accompanist, the Info Phil allowed computer and human to share the responsibility for coordinating their parts instead of the player being solely responsible. While the Info Phil makes things easier for the performer, the composer had mixed feelings about the adaptation. The majority of *Garden of Love* is characterized by a relentless, mechanical "groove" created from synthesized midi sixteenth notes or cut-up voice samples. Since mechanical precision is important for the character of the piece, the flexibility allowed by the Info Phil may be somewhat at odds with musical intent. We will return to discussing this issue in Section 4.

3.2 Billie

Billie, for alto saxophone and tape, incorporates a similar "voice-melody" approach to *Garden of Love* using samples from rare interviews with jazz singer Billie Holiday. However, since the piece is jazz-inspired and more relaxed, there was more room for tempo variation than in *Garden of Love*. We found that in *Billie*, freedom with timing led to new expressive possibilities. Frigo observed that playing with the adapted version gave her the impression of a "real-time conversation" with Billie because she could "flex the soundtrack in response to what she was saying." Frigo was also able to change the emotional impact in certain sections of the piece. She stated that choosing slightly different tempos from the original can "change the character of Billie's voice. It can create more anxiety, it can make the speech sound more laid back, or nonchalant, stressed..." JacobTV further commented that "with this technique, you could be a stage director, telling Billie to get more excited, or irritated, or angry . . . this person, Billie Holiday, comes alive through the speed Connie picks for the piece."

The character shift described by JacobTV and Frigo is clear when comparing the two different ways Frigo performed measures 21-26 of *Billie* at the NASA conference. The audio at `https://drive.google.com/file/d/1SFzXUInAHG9aukAuUxovLZzcBd6M3aeF/view?usp=sharing` shows Frigo first playing this section with the fix track, then with the Info Phil. The version with the original track, constricted by a fixed tempo, is calm, steady and deliberate. In the flexible version, Frigo chose to first accelerate, creating a sense of agitation, then slow down at the end of the phrase to give a sense of release. A link to her full performance and our talk at the NASA Biennial Conference can be seen at `https://vimeo.com/407549620`. Frigo preceded her "real" performance at the conference with a rather wild interpretation, demonstrating the high degree of flexibility supported by the Info Phil, though also raising questions about how the technology should work in tandem with musical considerations.

While performer and composer were both positive about

the performance at the conference, we did note some skepticism from at least one conference participant regarding the musical appropriateness of making the piece more flexible. *Bille* is well-known to the saxophone community, and the accompaniment recording released with the piece has enforced a high degree of conformity in the way it is performed since one cannot stray from the tape track's timing. We wondered if the preference by some for a more mechanical interpretation of *Billie* was simply the gravitational pull of what is familiar.

3.3 Farewell Feathered Friends

At its best, Info Phil adaptions allowed for the tight synchronization required by JacobTV's works without constricting the player to a fixed temporal interpretation. This advantage was most striking in our third adaptation, *Farewell Feathered Friends* for piccolo and tape. The tape track of *Farewell Feathered Friends* prominently features birdsong from seven species of endangered European birds. The piccolo often sounds much like another bird, in either conversation or in synchrony with the recorded birdsong, though perhaps more lyrical. The Info Phil version was performed by JSoM student Jamey Guzman, who enjoyed how the timing freedom in this version allowed her to respond to the birdsong. She commented in her recital introduction that "[the Info Phil] truly lets me lead and express the emotions in the piece without feeling locked into the same tempo every time like in a traditional fixed media piece." Guzman's performance can be viewed at `https://drive.google.com/file/d/1RbgunOAdcGy2oX9c8S0CyMQOTVh0ha5A/view?usp=sharing`.

4. OUR EXPERIENCE WITH JACOBTV'S MUSIC

Through our experience creating and working with the Info Phil adaptions for *Garden of Love*, *Farewell Feathered Friends*, and *Billie*, we have gained a deeper understanding of the advantages and questions raised by this new musical medium. Our observations coalesced into two main issues to consider when deciding among performance options for a tape piece.

4.1 Longevity

One of the biggest barriers to wide performance of live media pieces is the complexity needed to set up and perform live electronics. Often, the composer, or another individual familiar with the work's construction, must be personally involved in the performance, as expressed in Bruce Pennycook's article, "Who will turn the knobs when I die?" [2]. Using the Info Phil to control an originally fixed track offers the interactivity associated with live media with comparatively easy setup. Our work with Guzman and *Farewell Feathered Friends* is a good example. As a result of the COVID-19 pandemic, we were unable to offer Guzman in-person technical support. We only met once to give her necessary equipment: a pair of speakers and a Mac computer installed with the Info Phil software. Despite these less-than-optimal circumstances, Guzman was able to successfully run the program by herself during both rehearsals and her performance. As one can see in video of Guzman's performance in Section 3.3 , Info Phil usage is very straightforward; once the program starts, no external input from the performer or a technician is necessary. That said, the traditional non-interactive performance strategy of mixed music with fixed tape is nearly unbeatable in terms of facilitating a composition's longevity.

4.2 Learning vs. Performance

We observed that the performer's experience with the Info Phil could be divided into two categories: the *learning* process and the *performing* process. The players we collaborated with had a range of familiarity with JacobTV's music. Some, like Frigo, had performed the fixed media version of *Billie* for more than a decade. Others, like Taylor and Guzman, had just started their experiences with *Garden of Love* and *Farewell Feathered Friends*. All these collaborators brought engaging perspectives on their experiences performing and learning these works with the Info Phil.

4.2.1 The Info Phil as a Learning Tool

Taylor first started learning *Garden of Love* with the fixed track two weeks before trying the Info Phil version. He observed that the ability to practice sections with the accompaniment at a slower-than-marked tempo was a major benefit. Specifically, a slower tempo allowed him to pay attention to technical aspects of playing like tuning and rhythmic precision. He noted that "much like putting a piece together with a piano, you always start slow so you can find alignment between both parts, tuning issues, and work on passing off melodic lines ... Being able to take sections at slower tempi [with the Info Phil] allowed for me to work on the same things." Aside from these technical improvements, Taylor also gained new insight on *Garden of Love*'s musicality, finding "grooves in sections [he] was not previously aware of." Lastly, Taylor found that the flexible version of *Garden of Love* allowed him to gain a deeper understanding of the relationship between the solo and accompaniment parts:

> "Another advantage of being able to play through Garden of Love at a slower tempo was that it helped ingrain different parts of the track in my head. I walked out of the office that day with a much better idea of how the piece was constructed, how it was supposed to sound, and what aspects of the track were helpful to me as a performer. I definitely would say I learned more about the piece in those two hours than I did in the 2-3 week window prior in which I was learning it [with the fixed track]."

Thus a big potential benefit of the accompaniment system is in *learning* the piece at hand, whether or not one chooses to perform that way.

Having premiered Billie in 2003, Frigo had a different perspective on the learning process. Her experience with

the Info Phil version involved adjusting previous expectations playing with fixed track to work with the new, flexible format. Frigo was concerned saxophonists who only used the Info Phil to learn Billie could use the flexibility as a "crutch." When playing fixed media pieces, even rehearsing the work requires one to be intimately aware of how the solo part fits within the accompaniment track. But since the Info Phil is resilient to rhythmic mistakes, one can play the piece from start to finish without a robust understanding of the accompaniment. When asked whether one should learn from the fixed track or the flexible version, Frigo stated:

> "The reason I can be so flexible, even when experimenting, is because I know the soundtrack so well. And if you don't know the soundtrack, you're not actually aware of what you're playing with. When you play with fixed [track], you get to know there's a note right here, there's a cymbal here, a bass note here ... and if you don't know that, it's just like going into a rehearsal with a pianist without doing score study in advance .... [First practicing with the fixed track] would be a requirement just to give justice to the piece and understand how to create spontaneity within it."

In summary, Taylor noted a contribution the Info Phil brings to *learning* new pieces, though Frigo expressed some skepticism that this contribution could be abused. Of course, the fixed-media approach — playing along with a recording, is a time-honored way of learning the interrelations between parts, as well as introducing young musicians to the coordination demands of ensemble playing. However, we note an important problem with learning from fixed recordings. When the live player starts to lose track of her place in the larger structure, things often fall completely apart, with the live player unable to find her way back. In contrast, the Info Phil is more tolerant, able to help the player recover from a moment of uncertainty without getting irretrievably lost, potentially leading to a more robust experience.

4.2.2 Performing with the Info Phil

We consistently observed that the Info Phil makes the process of performing a fixed media piece significantly easier. The program is an equal partner in creating a synchronous performance, rather than putting this burden entirely on the soloist. For instance, the program allows a wind player to breathe in a more natural way, and supports, rather than punishes, the temporary glitches that pervade nearly all human performance scenarios.

On a deeper level, it solves the biggest challenge of playing with fixed media: making the performance sound spontaneous and interactive instead of constricted by a predetermined set of timing requirements. Though a skilled performer can create the impression of interactivity within a fixed tape framework, considerable virtuosity and familiarity with the work is necessary to sustain that illusion [2]. Even if one successfully projects the illusion of freedom with a non-interactive accompaniment, it can still make the performer feel like a puppet, tightly controlled by external strings. By fostering the performer's feeling of immersion and flow, the Info Phil can contribute to the their overall sense of comfort.

But are all types of music suitable to this flexible format? The Info Phil was originally conceived with the Romantic solo work in mind. Here the soloist may *lead* a pianist or ensemble using flexible timing, among other means, to make the music expressive. This leader-follower model is a reasonable approximation for a lot of solo music, while the recognition of these asymmetric roles is commonly acknowledged in discussions about music performance. However, from a different perspective, the leader-follower model seems even more justifiable in a human-machine musical collaboration since only one of the two performers actually cares what happens, or is fit to judge the level of success. Of the pieces we adapted, not all made these same implicit assumptions of fluid timing and leader-follower roles. Most notably, *Garden of Love*'s driving, metronomic tone felt at odds with the flexibility afforded by the Info Phil. Since rigidity is a key musical component, the added flexibility, while making the performance easier for the human, may work against the piece's aesthetic nature.

Of course there are a great many pieces of music that seem to require an unrelenting steadiness on the part of the performers, such as marches and perpetual motion pieces. Even if one accepts that inflexible timing is indeed the musical objective, there is still some subtlety in understanding the appropriateness of a flexible accompaniment system like the Info Phil. The accompaniment system does not keep the player from performing with robotic precision, so one can still pursue this inflexible musical agenda if desired. On the positive side in such a case: since the player, being human, will not succeed in being perfectly accurate, the accompaniment system will accommodate the small and inevitable inconsistencies. On the negative side, there will always be small inaccuracies in the system's identification of note onset times played by human musicians, thus drawing the accompaniment away from metronomic performance. The tension between these two issues creates a subtle tradeoff that, ultimately, must be decided by personal preference.

One of our subjects, Taylor, had played *Garden of Love* with both fixed and flexible tracks, thus was poised to make comparisons. He mentioned that while practicing with the fixed track, certain entrances were very stressful because an initial alignment misjudgement on his end could compromise the rest of the section. When playing with Info Phil, he was able to start sections without worrying about alignment issues. Energy not spent anticipating a tricky entrance can be instead focused on expressive or stylistic playing; Taylor commented that "this security allowed me to focus on playing my part at a higher level".

Even when tight synchrony between soloist and accompaniment is a decidedly good thing, as we believe to be the case in the overwhelming majority of Western solo music,

the issue still has its subtle side. While synchrony may be a desirable objective, it is not the only objective. A good accompanist seeks to preserve the internal consistency in her part, while, at the same time, staying with the soloist. These objectives often come into tension and cannot be resolved in any obvious way, though good musicians seem to successfully navigate this challenge all the time. We do not have a ready answer to this issue, except to observe that the Info Phil is among the first generation of accompaniment systems. We cannot resist recommending this problem as an unsolved challenge for the future development of accompaniment systems, since we certainly hope that the Info Phil will not be the last word on the subject.

5. GENERALIZING BEYOND FIXED-MEDIA COMPOSITIONS

Though the examples we investigated in this paper all involve adapting fixed-media works, e.g. recordings of "accompaniment," Info Phil technology can be extended to form a more general-purpose framework for new compositions using acoustic instruments and computer-generated sound. Here our presumption is that the live player would *lead* the performance, introducing an element of spontaneity and musicality sometimes missing from computer-only music. However, we still leave open limitless possibilities for both computer-generated sound and live processing/modification of the soloist's audio. In this way we hope to allow both human and computer to do what they do best. Ideas along these lines have been pursued at IRCAM, as with Manoury's works such as *Tensio* [6], though we seek to establish a portable implementation that would allow anyone to write or play such a piece without requiring a detailed knowledge of our system's inner workings.

The score to such a composition would be similar to a traditional Western score in some fundamental ways. The solo part would be notated with traditional rhythm and pitch, with a best approximation given for extended techniques like multiphonics, flutter-tonguing, quarter tones, etc. that do not fit neatly into traditional pitched notation. Notation whose timing might be represented in seconds could easily be translated into a traditional rhythmic scheme as well. We grant that there may be some musical elements in the solo part that are awkward to represent in traditional notation. However, the Info Phil understands its flexible timing as the warping of traditional rhythm through a time-varying tempo process, so we are bound to this rhythmic representation. Furthermore, we expect that the downside of using traditional notation for the solo part is just that it over-specifies musical notions that are not so precisely conceived, e.g. music without pulse naturally expressed in seconds.

Like the soloist's part, the computer's part is also represented in terms of traditional rhythm. As described in Section 2.2, it must clearly show all points of coincidence with the soloist as well as any other intricate rhythmic interrelationships . However, the important issue of what the computer will *do* at each rhythmic location is left open – what happens at these points will be "filled in" by the composer. Every rhythmic event in the computer part could be assigned to one or several actions, where potential actions include triggering playback of pre-recorded audio; recording soloist audio; processing soloist audio via filtering, harmonizing etc.; or terminating any of these processes. Ultimately, the computer's part is simply a list of actions that must be run when the associated musical times occur.

How is this list of musical actions executed? Just like performing a fixed accompaniment, the Info Phil listens to the traditionally-represented solo part, predicting solo note onset times as the performance evolves. Using these predictions, Info Phil creates a "scheduler" which continually updates the predicted times of future accompaniment events. Specifically, at any point during the performance the Info Phil focuses on the *currently pending* computer event — the first as-of-yet unperformed computer action. As solo note information is accumulated, this pending event will be scheduled and rescheduled, making use of all currently available information. To see the scheduling process in action, see the Lalo video referenced in Section 2. In this example and JacobTV's fixed media pieces, these predicted events are used to determine the amount of time stretching necessary to stay aligned with the soloist. However, in the context of the proposed live media system, the predicted time of each pending event will be used to directly trigger musical actions in the computer part. In practice, the program that implements the computer's part simply receives the scheduled and rescheduled event times from the Info Phil, acting when the currently-scheduled time finally arrives. A visual representation of how this scheduling mechanism could work in tandem with live media processing can be seen in Figure 2.

It is worth contrasting this paradigm for acoustic-computer compositions with the frequently encountered one in which an operator — often the composer themselves — must perform an intricate, and sometimes terrifying, sequence of button presses, slider movements, etc. in a virtuosic dance with the live performer. We seek to simplify the performance of such a piece to a more generic and less human-dependent model, thus making a composition more portable and potentially longer-lasting. We have not yet implemented this scheme, but envision that we could create an integrated Max patch encapsulating the listening and scheduling functionality of the Info Phil. This would eliminate potential communication lag between the Info Phil and Max/MSP, as well as simplifying inter-process communication.

Besides live composition, this general workflow can also apply to other domains where events are scheduled based on a pre-determined score. Possibilities include opera supertitles, or automatically controlling stage lighting and projected backgrounds for performances with audio and visual components.

We were able to prototype this live media triggering pipeline by adapting the music video for *Billie* to work with the Info Phil. During *Billie* performances with the original tape track, players have the option of simultaneously projecting a video where visual changes align with specific beats. Since our flexible renditions of *Billie* deviate from the tape's pre-determined tempos

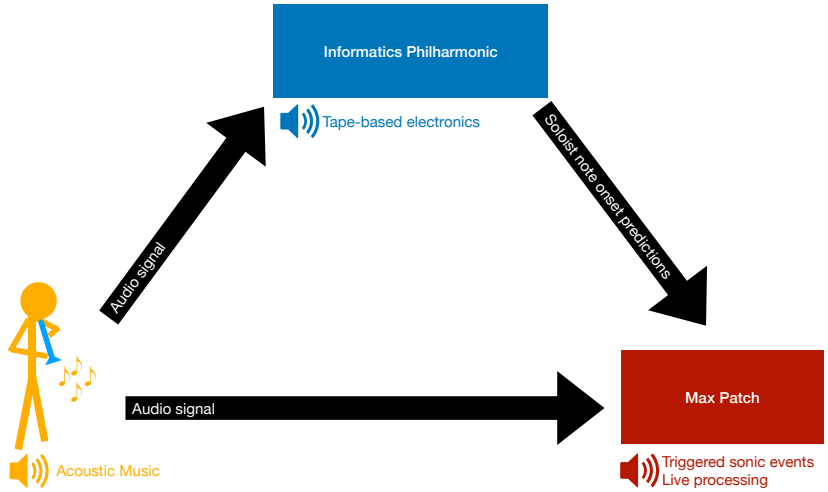


Figure 2: Schematic of information flow in pieces combining acoustic soloist, flexible tape, and live processing. Audio data from the soloist is first processed by the Info Phil to produce flexible, tape-based electronics. The Info Phil then sends times of projected accompaniment events to an external Max patch. The Max patch can serve the dual purpose of both directly processing soloist audio and triggering changes based on the player's position in the score.

and rhythms, the original video could no longer be used during performance. A new way of displaying video was needed to match the player's real-time decisions. We created a responsive video by linking the video's playback rate to scheduling information from the Info Phil. That is, the Info Phil sends messages in the form "player will reach score position x at time y" to a Max patch. When the Max patch receives such a message it resets its video playback rate so that the musical position (beats) will be reached at the desired time (seconds). A demo can be seen at `https://drive.google.com/file/d/1fq_XTW6pg-ywjt8NWTIG1cDetNCu2irH/view?usp=sharing`.

Our last word is a call to composers to write the kind of piece that can be implemented in the manner we describe. We invite such collaborations and look forward to seeing them take shape in the future.

Acknowledgments

We thank JacobTV, Connie Frigo, Jamey Guzman, and Wes Taylor for their invaluable contributions to this project.

6. REFERENCES

[1] K. Stockhausen and J. Kohl, "Electroacoustic Performance Practice," *Perspectives of New Music*, vol. 34, no. 1, pp. 74–105, 1996, publisher: Perspectives of New Music. [Online]. Available: http://www.jstor.org/stable/833486

[2] B. Pennycook, "Who will turn the knobs when I die?" *Organised Sound*, vol. 13, no. 3, pp. 199–208, Dec. 2008. [Online]. Available: https://www.cambridge.org/core/product/identifier/S1355771808000290/type/journal_article

[3] R. Dannenberg, "An on-line algorithm for real-time accompaniment," in *Proceedings of the International Conference on Computer Music (ICMC)*, 1984, pp. 193–198.

[4] B. Vercoe, "The synthetic performer in the context of live performance," in *Proceedings of the International Conference on Computer Music (ICMC)*, 1984, pp. 199–200.

[5] A. Cont, "ANTESCOFO: Anticipatory Synchronization and Control of Interactive Parameters in Computer Music." in *International Computer Music Conference (ICMC)*, 2008, pp. 33–40.

[6] P. Manoury, "Compositional Procedures in Tensio," *Contemporary Music Review*, vol. 32, no. 1, pp. 61–97, Feb. 2013. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/07494467.2013.774514

[7] C. Raphael, "Music Plus One and Machine Learning," in *Proceedings of the Twenty-Seventh International Conference on Machine Learning*, 2010, pp. 21–28.

[8] N. M. Collins, "Towards Autonomous Agents for Live Computer Music: Realtime Machine Listening and Interactive Music Systems," Ph.D. dissertation, University of Cambridge, Centre for Music and Science, 2006.

[9] C. Raphael, "Jan beran: Mist covered mountains and winter711," http://www.music.informatics.indiana.edu/~craphael/music_plus_one/jan_beran.html, 2005, [Online; accessed 11-March-2021].

CONVOLUTIONAL NETWORKS FOR VISUAL ONSET DETECTION IN THE CONTEXT OF BOWED STRING INSTRUMENT PERFORMANCES

Grigoris BASTAS (Γρηγόρης Μπάστας)[1,3], **Aggelos GKIOKAS** (Άγγελος Γκιόκας)[2],
Vassilis KATSOUROS (Βασίλης Κατσούρος)[1], and **Petros MARAGOS** (Πέτρος Μαραγκός)[3]

[1]*Institute for Language and Speech Processing (ILSP)*, **Athena R.C.**, Athens, Greece
[2]*Music Technology Group*, **Universitat Pompeu Fabra**, Barcelona, Spain
[3]*School of Electrical and Computer Engineering*, **National Technical University of Athens**, Athens, Greece

ABSTRACT

In this work, we employ deep learning methods for visual onset detection. We focus on live music performances involving bowed string instruments. In this context, we take as a source of meaningful information the sequence of movements of the performers' body and especially the bowing motion of the (right) hand. Body skeletons for each video frame are extracted through OpenPose and are then used as input for Temporal Convolutional Neural Networks (TCNs). TCNs prove capable of handling such temporal information by conditioning outputs on an adequately long history (i.e. variable receptive field), ensuring highly parallelizable lightweight computations and a multitude of trainable parameters that provide robustness. As another source of information for our task, we consider the more subtle movements of the (left) hand fingers which are responsible for pitch changes. Detections in this case rely directly on pixel data from specifically chosen regions of interest. Here, a 2D Convolutional Neural Network (CNN) is applied on the input in order to learn the features to be fed to the TCN. The models were trained and evaluated on single-player string recordings from the University of Rochester Multi-Modal Music Performance (URMP) Dataset. We show that these two approaches provide some complementary information.

1. INTRODUCTION

Onset times is one of the most fundamental elements of the temporal organization of a music piece. Onsets are placed relatively to the metrical grid of a music piece and their positions on this grid (along with duration and velocity) greatly determines the rhythm structure of a music performance. Moreover, the micro-timings, i.e. small time deviations of the onset with respect to the ideal metrical grid are very related to aspects of human expressivity. Consequently, onset detection comprises a fundamental problem in the field of Music Information Retrieval and it is related to many other tasks including tempo estimation, beat tracking, music transcription, source association.

Music is often experienced by humans in a visual context. The visual perception plays a key role whether the stimulus involves an album cover, a video clip or a live music performance [1]. Research in psychology has documented an impact of the visual information on the judgment over musical live performances [2] or even on how musical structure is perceived [3]. During the past few years, audiovisual analysis has drawn the attention of the music information retrieval community. Visual information can be important for deducing a performer's stylistic techniques, recognizing the playing instruments, capturing the emotional variations in a piece, etc. Innovative information extraction techniques employed in such a multimodal context have been evolved [4]. Traditional image processing, pattern recognition and deep learning methods have been used to deal with tasks relevant to this emerging field.

The development of convolutional networks has been decisive for the advancements in computer vision during the last decades. Convolutional Neural Networks (CNNs), which apply two-dimensional convolutions, play a crucial role in machine learning because they enable learning latent features from images, adaptable according to each specific task. During the last few years, convolutions have also been employed to handle sequential data using learnable filters to convolve over the axis of time. These types of architectures which involve one-dimensional convolutions are called Temporal Convolutional Networks (TCNs) and have exhibited some advantages over the often employed Recurrent Neural Networks (RNNs) such as Long Short-Term Memory (LSTM) units.

For many musical instruments, the produced sounds correspond to certain visible movements and specific positioning of the instrument player's hands. More particularly, with regard to the bowed string instruments, the bowing motions lead to the articulations of music notes (see Fig. 1). Such visual cues are detectable by the human eye quite easily. Pitch change requires altering the positioning of the fingers on the neck of the instrument. Each fingering transition is strongly correlated to note onsets.

Capturing such visual information content with computational methods can be challenging, that is why state-of-the-art computer vision techniques need to be employed. Occlusions and irrelevant or subtle movements are not easy to cope with. Naturally, not all right hand movements correspond to onsets, and several onsets can be produced by legatos (i.e. left hand pitch change) without changing the

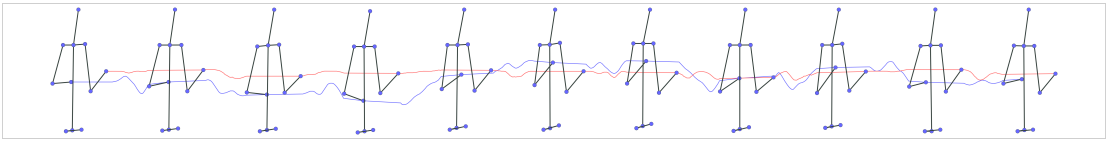


Figure 1. Instances of a moving skeleton extracted from a violin performance. Trajectories of the right hand are displayed with blue color and for the left hand with red.

direction of the bowing motion. Similarly, fingering transitions of the left hand may not bear musical information, or might not signal the exact time location of an onset. Such motion can also be difficult to discriminate from vibrato. Also, onsets of the same note can be produced simply by new bow strokes without any change in the fingering. Difficulties arise since part of the left hand is occluded by the neck of the instrument and occlusions might occur between fingers, depending also on the relative position of the hand to the camera. Additionally, discriminating between the motion of the hand in the scene and the relative motion of the hand on the neck of the instrument constitutes another challenging aspect of the problem. Since these movements can be very subtle, the stability of the camera and its distance from the hand are also crucial parameters.

The standard process to infer onset locations is by working on the audio signal. In this paper, however, we present a method for visual onset detection. We deploy TCNs and CNNs, and we demonstrate that the visual modality can be a source of meaningful musical information that, handled correctly, can help to cope with this challenge. We focus on bowed string single-instrument performances where the hand and body movement can provide visual information for the onset locations.

The rest of the paper is organized as follows. Section 2 provides related work for the fields of onset detection, the multimodal or purely visual approaches for music analysis and generation. Section 3 describes the proposed method, and section 4 provides the experimental results. Section 5 concludes the paper with discussion and future directions.

2. RELATED WORK

2.1 Visual-based Music Analysis and Generation

Visual-based approaches on music analysis have been applied on several different tasks during the past years. The visual modality has played a critical role in tasks such as audio-visual source association, fingering analysis, playing/non-playing (P/NP) activity detection, vibrato analysis, automatic music transcription and onset detection [4]. Parekh et al. [5] have engaged in audio-visual source separation in polyphonic performances, focusing on motion tracking of bowed string performers, using unsupervised learning techniques like Non-negative Matrix Factorization (NMF). Several research teams have worked on fingering tracking focusing on instruments like guitar, based on computer vision and statistical tools [6–9], aiming as far as music transcription in symbolic forms like tablature [10, 11]. In [8], finger tracking on the guitar player was used to detect "key frames" (i.e. the time locations of chord changes), a notion very close to onset-corresponding frames. In [12] a method for guitar transcription is presented relying on video close-up recordings of the vibrating strings [12]. Fingering recognition and hand tracking systems have also been developed for piano performances [13, 14] and violin [15].

Other teams have undertaken audio-visual analysis using deep learning models. CNNs have been used in tasks like instrument recognition where the visual modality is prominent [16, 17]. CNNs have also been used for localization of specific regions in video frames that correspond to distinct music signal sources and thus also make it possible to separate the two signals [18]. Extracted skeleton poses have also been used to extract music information from the visual modality or to study audio-visual correspondence. Pedersoli and Goto [19] introduced the task of Dance Beat Tracking, where they employed TCNs to predict onset locations, having as input skeleton poses of the dancers while performing.

Apart from the context of music analysis of the visual content, body motion related recognition techniques have been deployed in the context of music generation, as for example using skeleton data from Kinect sensor for air-guitar playing [20] or using finger motion data for recognizing gestures in order to perform on virtual instruments [21]. GANs were recently been employed for visually enhanced audio inpainting based on live music performances [22].

2.2 Audio and Visual Onset Detection

Traditionally, the task of onset detection has been dealt with using information from the audio signal. The state-of-the-art uses a Convolutional Neural Network (CNN) architecture [23] in a similar way as CNNs have been used for edge detection in the field of computer vision. Three versions of 80-band log-mel features were used to represent the audio input with three different corresponding window sizes. Hence, the input was given in the form of three channels on which 2D convolutions were applied using two layers involving max-pooling. RNNs have also been employed successfully to tackle the problem [24].

Zhang and Wang [15] engaged in audio-visual music transcription for violin. Onset detection was a fundamental part of such a system. They presented both visual and audio methods for detecting onset times. The audio-based method relied on training Gaussian Mixture Models (GMMs) on Mel-Frequency Cepstral Coefficient (MFCC) features. The visual-based method was centered on the analysis of two different sources of visual information provided by two cameras, one recording the bowing motion

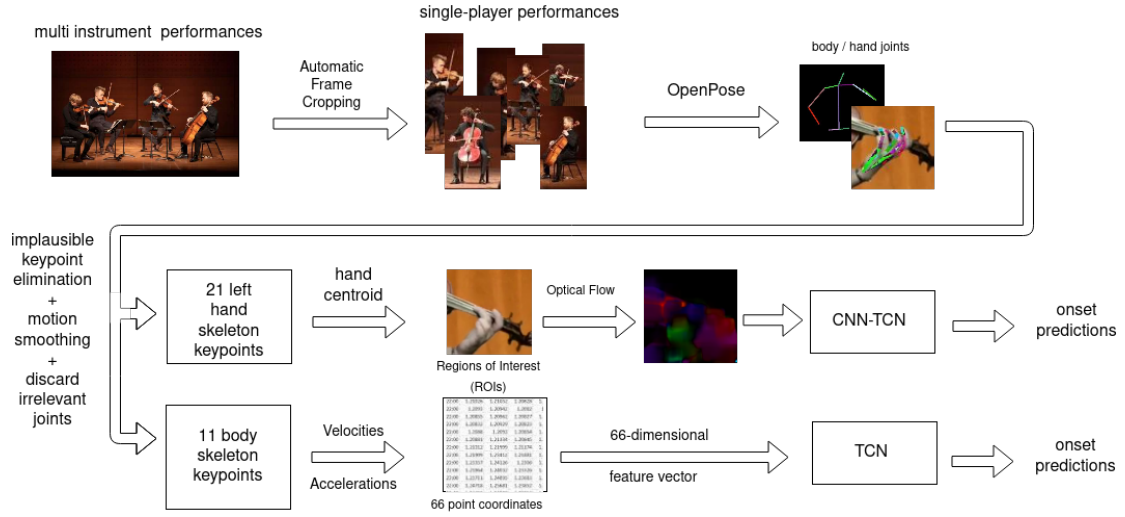


Figure 2. A flow diagram depicting our methodology for visual onset detection.

and the other recording the fingering transitions. Distinct prediction functions for the two video signals were employed, in the first case relying on tracking the hand holding the bow and in the second, concentrating on recognizing the violin strings and the relative position of the hands on them. The resulting prediction scores of each distinct source of information were finally combined using feature level and decision level fusion.

Audio-visual analysis focusing on onset detection for string ensembles has also been conducted by Li et al. [25], serving as a basis for score-informed audio-visual source association. The onset locations were estimated by focusing on bow stroke detection and more specifically, on the zero-crossings of the principal motion velocity, computed using the information by optical flow vectors. Audio-visual source association has been handled using vibrato analysis [26], too. In contrast with the aforementioned works, in [27], the visual information was reduced, using OpenPose [28], to keypoints representing body and finger joints. The vibrato and bow stroke approaches have been combined and the onset detection task has been aided by following the finger movements of the players' left-hand, thus also permitting the generalization of the analysis on woodwind and brass instruments. Recently, TCNs have been used to detect onsets from the audio and visual sources and for fusing these two modalities [29]. In our current work, we focus on the visual modality and present a significantly improved feature extraction procedure and a more robust TCN model architecture to process skeleton data. We compare this strategy to a pixel-based approach which aims at capturing left-hand motion.

3. METHOD DESCRIPTION

3.1 Method Overview

An overview of the proposed methods is illustrated in Fig. 2. First, the videos are cropped in order to work with single-player performances. For each single-player performance, we apply OpenPose in order to extract the performer's skeleton. The extracted skeleton is smoothed and we subsequently extract velocity and acceleration features, while from the centroid of the left-hand keypoints we extract Regions of Interest (ROIs) in order to isolate the hand from the rest of the image. For these ROIs, we compute the optical flow to capture the motion of the left hand. The extracted body skeleton features are used to train a TCN, while the left-hand features are used to train a network that comprises conventional CNN and TCN layers.

3.2 Video Processing

3.2.1 Preprocessing

As a first step, the videos were automatically cropped using ffmpeg command-line tool, in order to ensure that only one person would appear in each recording. For this task, we took advantage of the available information about the number of the players involved in every performance and we chose to segment the frames accordingly, in equal parts, with respect to the x-axis. Minor corrections were required to be done by hand in the case of only one recording. We used OpenPose [30] for 2D pose estimation by employing the BODY_25 output format with hand keypoint detection enabled using the officially suggested scale number and scale range values (6 and 0.4 respectively) for achieving best results. We thus extracted body skeletons comprised of 25 points, together with 21 extra keypoints for each hand.

3.2.2 Body Motion Information

Deriving onsets only from body movements does not require all of the predicted skeleton keypoints. In this setting, hand joint keypoints, ears and eyes, as well as those points corresponding to the knees and below were all ignored because they were considered redundant or in some cases they were prone to occlusions. We were left with 11 body

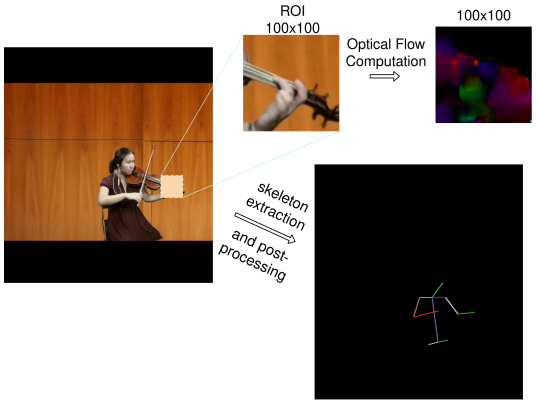


Figure 3. Skeletons and ROIs are extracted from a video frame involving a violonist from a performance in the URMP Dataset [28]. Optical flow is further computed for the isolated hand ROI.

keypoints (see Fig. 3) and hence 22 coordinate values. We followed the processing steps from [31] in order to create continuous skeletons: we eliminated joints with confidence score lower than 0.2 and, in order to rule out abrupt and unnatural keypoint shifts, we discarded keypoints if distance of a joint from one frame to the other was found larger than the 10 percent of the nose to hip distance. In frames where certain joints were occluded or eliminated (following the above criteria), or even in the rare cases where OpenPose failed to capture a skeleton, the keypoints were recreated using linear interpolation between valid frame instances. Centered moving average with window size 5 was used to further smooth the skeleton motion.

After having computed the centroid of the skeletons for each frame and the corresponding standard deviation of the keypoints, we have applied z-score normalization in 2D space using the mean values of the two magnitudes along time. This procedure was applied separately for each performance. In this manner, we have first of all imposed a new center of axis which enabled us to rule out the differences among the positioning of the performer in relation to the edges of the video frames for each distinct recording. Secondly, we eliminated the variation of skeleton sizes among the performances which is introduced especially due to the different distances of the performer from the camera. Finally, keypoint velocities and accelerations were used as additional features, leading to a 66-dimensional input.

3.2.3 Pixel Data

In order to get information from the hand responsible for the pitch changes, one should focus on the finger movements and positioning. The OpenPose keypoints corresponding to the performers' hands are highly inaccurate in this particular setting since they are often occluded by the neck of the instrument and involve jitter. The relatively long distance of the hand from the camera makes things even worse. This raises the need for extracting features directly from the images. For this task we chose to employ Convolutional Neural Networks (CNNs). In order to eliminate redundant visual information and focus on the (left) hand movements we took advantage of the predicted hand keypoints to isolate specific regions of interest (ROIs). To achieve this, we computed the centroids of the left hand joint for each consecutive frame as in [27], but we used only the average centroid across time. This point in 2D space serves as the center of a steady square ROI. We avoided the use of moving ROIs following the hand centroid as in [27] because we observed a strong effect of the changing background patterns to our recognition system. The size of the ROIs in the case of violin and viola performances was 100x100 pixels while for the violoncellos and the double basses 200x200 frames in order to capture all the range of vertical hand movement on the instrument's neck. These last larger ROIs were rescaled to 100x100 images in order to ensure same input sizes for the onset detector. When dealing with ROIs that extend out of the frame (on the right side to be specific), we employ zero-padding to ensure proper image sizes.

3.3 Temporal Convolutional Networks

Temporal Convolutional Neural Networks (TCNs) constitute a family of architectures designed to grasp dependencies among sequential data. Dilated one-dimensional convolutions enable control of the network's receptive field, that is the considered length of dependencies among temporal data. Every added layer increases the temporal scope of the network exponentially. Consequently, with few added trainable parameters one can ensure adequate source of information. Another hyper-parameter that is pertinent to the temporal dependencies that one intends to grasp is the size of the convolutional filters across the time axis. A good combination of these two values permits handling of complex data associations. As in the case of two-dimensional convolutions, the computations can be efficiently parallelized using GPUs. That is a limit that Recurrent Neural Networks (RNNs) are confronted with since each new forward pass shall wait for the output of the previous step to be produced. Various versions of TCNs have recently been introduced in [32–34], for generative or classification purposes.

Henceforth, TCNs definitely fit our needs for the problem of visual onset detection which requires handling sequential data. TCNs proved to lead to exceptional results with strong generalization abilities. Using a stack of TCNs, instead of only one, was found to lead to over-fitting in our setting which involves relatively few data. The architecture that was tested in this work is inspired by Wavenet [32] and the TCN proposed in [34]. We do not however stick to the proposed causal setting where each prediction relies on past observations. Our non-causal configuration visualized in Fig. 4 involves 9 layers ($l \in [1, 9]$) where two distinct sets of one-dimensional convolutional filters $W_{1,l}$ and $W_{2,l}$ are learned. Weight normalization is applied in both arrays. Hyperbolic tangent is used in the first case and sigmoid activation function in the other. The two outputs are then combined using element-wise multiplication.

This parallel configuration imposes non-linear filtering

$\tanh(W_{1,l} * x)$ and a learnable mask $\sigma(W_{2,l} * x)$ applied in each layer. In every layer, 256 convolutional filters comprise each of the two convolutional blocks. Residual connections appear in each layer involving 1x1 convolutions to upsample the input and fit it to the size of each layer's output when needed. The use of 9 layers with a variable dilation factor 2^l (assuming videos with standard frame rate (29.97 fps) as in our case) entails a receptive field of about 17 sec, centered around the timestamp for which a prediction is to be made. Small kernels of size 3 across the time axis are used and symmetrical zero-padding (increasing from layer to layer in accordance to the dilation factor) is applied at the beginning and at the end of each performance to ensure non-causality. Dropout with probability 0.25 is applied to avoid over-fitting, as well as gradient clipping with a corresponding coefficient of value 0.2. A linear fully connected layer followed by a softmax function is used to output 2-dimensional vectors for each input frame, with each coordinate representing the probability of an occurring and a non-occurring onset respectively.

3.4 Convolutional Neural Networks

However suitable TCNs may be when fed with features such as sequences of post-processed skeleton coordinates, they cannot efficiently extract information directly from pixel data as in the case of left-hand ROIs. Proper features should be extracted first in this case. An end-to-end configuration where latent features are learned has been proven an effective strategy since it can be directly adaptable to specific tasks. CNNs constitute a standard for image handling. In our case, a 3-layer CNN is employed with 5x5 kernel size as displayed in Fig. 4. The number of filters increases from one layer l to the next, by $4 \cdot l$. ReLU is used as activation function after batch normalization. Max pooling with kernel size 3x3 and stride 3 is applied in order to downsample each layer's output. In the network's output, 16 feature arrays of size 4x4 are produced and are then fed to the TCN after having been flattened, ending up with vectors of 256 dimensions.

4. EXPERIMENTS AND RESULTS

4.1 Datasets

We trained and evaluated the proposed models on video recordings from the University of Rochester Multi-Modal Music Performance (URMP) Dataset [28]. This dataset provides videos of multi-instrument performances that were created by assembling audio-visual recordings of individual music players performing separately, yet coordinated. The audio recordings of individual instrumental performances are also provided in the dataset, thus enabling the matching of each separate track with the corresponding cropped video (i.e. individual performances). Among these videos only the string instrument performances were used in our experiments. So, in total 61 single-instrument performances comprise our data. The duration of these performances vary strongly from 35 sec to 3.8 min.

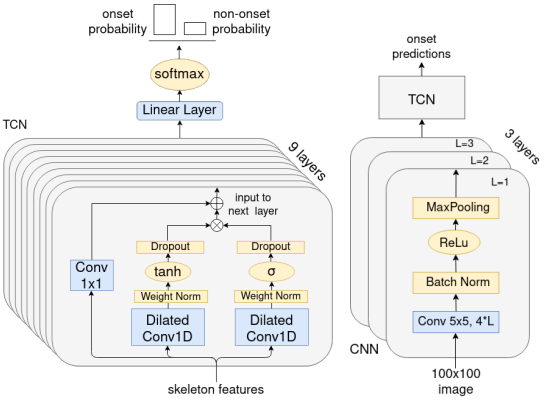


Figure 4. On the left side a 9-layer TCN is displayed, receiving as input skeleton features. A linear fully connected layer is applied on the output with a softmax activation function. On the right side a CNN-TCN model is depicted. The 3-layer CNN learns features from consecutive images and feeds them as input to a TCN.

4.2 Training and Evaluation Procedure

For the purpose of our models' evaluation we undertake 8-fold cross-validation using pytorch python library. The code can be found in [35]. The data were shuffled beforehand. All string instruments were involved in the training procedure. While training, a window of 3 frames around the annotated timestamps was considered to correspond to valid onset instances and, hence, we assigned as ground-truth for the 2D softmax output the probability vector $y = (1, 0)$ in the case of occurring onset and $y = (0, 1)$ when no onset occurs. Trainable parameters are considered to be all the 1D and 2D convolutional filters involved, plus the fully-connected linear layer in the output. Cross-entropy loss function was employed. The value 0.001 was opted for the initial learning rate, together with Adam optimizer. The average F measure is computed separately for the training, validation and test set. A maximum of 250 training epochs were run for each fold and the best model parameters were stored using as a criterion F measure values calculated for the validation set. After training, the best performing model versions in the validation set were then reloaded for the final testing. Two experiments were run:

- a TCN was applied on the extracted skeleton features
- a CNN-TCN was applied on the left-hand ROIs

An estimated note onset was considered to be correct if it was found ± 50 ms around the annotated timestamp. This is an adequately small range in the case of visual onset (also proposed in [4]), used instead of the $\pm$ 25 ms tight criterion, since the distance between subsequent video frames ($\sim$33 ms) exceeds this value. Local maxima of the activation function were computed. The peaks were found using a centered moving maximum with a window size of 3 consecutive frames. We used a threshold of 0.5 to derive predictions. For each individual recording, either audio or video, precision, recall and F measure were computed.

4.3 Results

4.3.1 Skeletons and Hand ROIs

As a first step, we investigated the ability to detect onsets by focusing on body movements. Sequences of post-processed 11-keypoint skeletons were given as input to the TCN model (TCN-Sk). Next, we applied the CNN-TCN (CNN-TCN-ROI) on the isolated ROIs capturing the left performers' hand. We compared the two results quantitatively and qualitatively. The model trained on the skeleton poses outperformed the pixel model by 17% in the overall accuracy as presented in Table 1. This fact leads us to the conclusion that the explicit movements of the body and especially the bowing motions of the right hand (see Fig. 1) can provide very clear information concerning the time of each note articulation, even on the relatively small training set we used. One can notice that Precision exceeds Recall for TCN-Sk. On the other hand, CNN-TCN-ROI yields slightly greater Recall than Precision. This signifies that TCN-Sk behaves in a more "reluctant" way as compared with CNN-TCN-ROI which takes more "risky" decisions, thus being prone to more false positive predictions. This can be interpreted by the fact that vibrato induces a lot of irrelevant motion which can be quite challenging for the corresponding classifier to discriminate from fingering transitions.

We also present the average results for each different instrument. Both models performed best in the case of violins where TCN-Sk reached the F measure value of 0.846. Both models exhibited their worst performance in the case of cellos. Each distinct model yields quite average results for double bass and viola. The relatively low results given by CNN-TCN-ROI in the case of cello performances can again be explained by the extensive use of vibrato by the cellist and the small area that the left hand occupies in the downscaled 100x100 frames. This in not the case for the double bass even if the same downscaling was forced on the extracted ROIs because, in all the three videos that this instrument appears, the position of the performer in each recording is closer to the camera than usual. Finally, the aforementioned pattern with regard to Precision and Recall holds true for both models in three out of four instruments.

4.3.2 Comparison with Previous Works

The above results can be compared with the performance of models presented in previous works. Bastas et al. [29] have deployed another TCN variant (with 6 layers) on post-processed upsampled skeleton data (93.75 fps). This frame rate was chosen to match the audio spectral representations which were themselves fed to a 4-layer TCN dedicated to the aural modality.

The current method outperforms the previous visual approach by 13.9% (see Table 2). There are various reasons for this. In the current setting we avoid standardization of the input vectors across time since it can lead to a loss of information with regard to the (relative) position of each keypoint. This is because the values of features corresponding to different coordinates of distinct body joints are forced to acquire a zero mean value across time. By avoiding standardization across time we also ensure intact ranges of motion of each keypoint, since we avoid the enforcement of a common standard deviation (i.e. $\sigma = 1$). What proved to be important is also the size of the receptive field which, in the previous method, is quite small (only 0.68 sec). Finally, we should also consider as a boosting factor the current configuration of the TCN-Sk which is more similar to the Wavenet architecture since it involves gated activation units. The results obtained by the pixel data are very promising as well. Even though the onset detection from fingers is more difficult, CNN-TCN-ROI yields results less than 4% worse than the results obtained from the skeletons with the previous method.

To obtain a more comprehensive view on the onset detection task, we compared our method to state-of-the-art methods applied on audio. The CNN model presented in [23] is trained on spectral representations from various audio excerpts. It is packaged in the madmom library and so it can be easily tested on the URMP Dataset. The TCN used in [29] is trained and tested using 8-fold cross-validation on the URMP Dataset. The results of TCN-Sk are less than 11% lower than the ones obtained by the CNN and 14.2% lower than the TCN that uses the aural modality as input. Finally, we notice a slightly greater standard deviation for the two newly introduced methods, which indicates better adaptation on a certain subset of the data, naturally on different instruments.

Instrument	Precision	Recall	F measure
Skeletons (TCN-Sk)			
Violin	0.867	0.833	0.846
Viola	0.785	0.722	0.746
Cello	0.655	0.596	0.620
Double Bass	0.730	0.734	0.732
Total	0.806	0.762	**0.779**
Hand ROIs (CNN-TCN-ROI)			
Violin	0.685	0.737	0.705
Viola	0.581	0.587	0.574
Cello	0.390	0.324	0.349
Double Bass	0.544	0.598	0.567
Total	0.604	0.622	**0.606**

Table 1. Precision, Recall and F measure results for the two proposed models (TCN-Sk and CNN-TCN-ROI). Separate average measurements for each instrument and total results after 8-fold cross-validation.

Model	F measure
TCN-Sk	0.779±0.079
CNN-TCN-ROI	0.606±0.092
TCN on Skeletons (Bastas et. al [29])	0.640±0.058
TCN on Audio (Bastas et. al [29])	0.921±0.018
CNN on Audio (Schlüter and Böck [23])	0.886±0.012

Table 2. Mean value and standard deviation of the F measure results for different tested methods on URMP [28].

Model	Recall
TCN-Sk	0.762
CNN-TCN-ROI	0.622
Combined True Positives	0.837

Table 3. Recall results of the separate models and of their combined yielded true positives.

4.3.3 Complementary Outputs

The question arises about whether these two methods capture complementary information or not. For this purpose, we assigned for each annotated onset a True label if it was indeed detected and False if it was not. We did this for all the string performances, for both the skeleton and the pixel model. The union of the Boolean labels outputed by the two models for each performance yields the combined true positive predictions. If the number of true positives or their proportionate occurrence (i.e. the Recall) would remain the same as the one of the best performing model, then there would not be any complementary information captured by the two approaches. However, the Recall value that results from the previous procedure is found to be higher by 7.5% than the Recall of TCN-Sk as depicted in Table 3. This finding is in agreement with the fact that note articulation might involve only fingering transitions (i.e. legatos) or only bow strokes (i.e. same note played). It also brings to light the distinct value and possibilities of each separate approach.

5. CONCLUSION

Both visual onset detection approaches (i.e. the one relying on skeleton features and the one relying on pixel data) proved successful in capturing information pertinent to this task. The subtlety of the fingering transitions on the instrument's neck is shown to pose greater difficulties for a model to grasp. However, the complementary information that can be captured with this method should be considered of great value for an enhancement of the overall performance on the task. Hence, as future work, one priority would be the development of a fusion method which would be able to efficiently combine the information captured from the two different information sources. Our preliminary experiments [29] on fusioning skeleton data with audio, pave the way for advancing new fusion methods. Multi-modal fusion involving trainable models was proved beneficial also in tasks like speech recognition [36]. One interesting direction would also be to study the possibility of an approach independent of a skeleton extraction phase. One reason for this is to reduce the time spent in the inference procedure by making predictions relying directly on the pixel data.

Acknowledgments

We would like to thank a lot our dear colleague Kosmas Kritsis for his help organizing some of the material and his valuable scientific comments.

6. REFERENCES

[1] F. Platz and R. Kopiez, "When the eye listens: A meta-analysis of how audio-visual presentation enhances the appreciation of music performance," *Music Perception: An Interdisciplinary Journal*, vol. 30, no. 1, pp. 71–83, 2012.

[2] C.-J. Tsay, "Sight over sound in the judgment of music performance," *Proceedings of the National Academy of Sciences*, vol. 110, no. 36, pp. 14 580–14 585, 2013.

[3] W. Thompson, P. Graham, and F. Russo, "Seeing music performance: Visual influences on perception and experience," *Semiotica*, vol. 2005, no. 156, pp. 203–227, 2005.

[4] Z. Duan, S. Essid, C. C. Liem, G. Richard, and G. Sharma, "Audiovisual analysis of music performances: Overview of an emerging field," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 63–73, 2018.

[5] S. Parekh, S. Essid, A. Ozerov, N. Q. Duong, P. Pérez, and G. Richard, "Motion informed audio source separation," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 6–10.

[6] A.-M. Burns and M. M. Wanderley, "Visual methods for the retrieval of guitarist fingering," in *Proceedings of the 2006 conference on New interfaces for musical expression*. Citeseer, 2006, pp. 196–199.

[7] C. Kerdvibulvech and H. Saito, "Vision-based guitarist fingering tracking using a bayesian classifier and particle filters," in *Pacific-rim symposium on image and video technology*. Springer, 2007, pp. 625–638.

[8] Z. Wang and J. Ohya, "Tracking the guitarist's fingers as well as recognizing pressed chords from a video sequence," *Electronic Imaging*, vol. 2016, no. 15, pp. 1–6, 2016.

[9] J. Scarr and R. Green, "Retrieval of guitarist fingering information using computer vision," in *2010 25th International Conference of Image and Vision Computing New Zealand*. IEEE, 2010, pp. 1–7.

[10] M. Paleari, B. Huet, A. Schutz, and D. Slock, "A multimodal approach to music transcription," in *2008 15th IEEE International Conference on Image Processing*. IEEE, 2008, pp. 93–96.

[11] B. Duke and A. Salgian, "Guitar tablature generation using computer vision," in *International Symposium on Visual Computing*. Springer, 2019, pp. 247–257.

[12] S. Goldstein and Y. Moses, "Guitar music transcription from silent video." in *BMVC*, 2018, p. 309.

[13] D. O. Gorodnichy and A. Yogeswaran, "Detection and tracking of pianist hands and fingers," in *The 3rd Canadian Conference on Computer and Robot Vision (CRV'06)*. IEEE, 2006, pp. 63–63.

[14] A. Oka and M. Hashimoto, "Marker-less piano fingering recognition using sequential depth images," in *The 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision*. IEEE, 2013, pp. 1–4.

[15] B. Zhang and Y. Wang, "Automatic music transcription using audio-visual fusion for violin practice in home environment," 2009.

[16] O. Slizovskaia, E. Gómez Gutiérrez, and G. Haro Ortega, "Correspondence between audio and visual deep models for musical instrument detection in video recordings," 2017.

[17] O. Slizovskaia, E. Gómez, and G. Haro, "Musical instrument recognition in user-generated videos using a multimodal convolutional neural network architecture," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, 2017, pp. 226–232.

[18] H. Zhao, C. Gan, A. Rouditchenko, C. Vondrick, J. McDermott, and A. Torralba, "The sound of pixels," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 570–586.

[19] F. Pedersoli and M. Goto, "Dance beat tracking from visual information alone."

[20] A. Zlatintsi, P. P. Filntisis, C. Garoufis, A. Tsiami, K. Kritsis, M. A. Kaliakatsos-Papakostas, A. Gkiokas, V. Katsouros, and P. Maragos, "A web-based real-time kinect application for gestural interaction with virtual musical instruments," in *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*, 2018, pp. 1–6.

[21] K. Kritsis, A. Gkiokas, M. Kaliakatsos-Papakostas, V. Katsouros, and A. Pikrakis, "Deployment of lstms for real-time hand gesture interaction of 3d virtual music instruments with a leap motion sensor," in *Proceeding of the 15th Sound and Music Computing Conference (SMC2018)*, 2018, pp. 331–338.

[22] H. Zhou, Z. Liu, X. Xu, P. Luo, and X. Wang, "Vision-infused deep audio inpainting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 283–292.

[23] J. Schlüter and S. Böck, "Musical onset detection with convolutional neural networks," in *6th international workshop on machine learning and music (MML), Prague, Czech Republic*, 2013.

[24] F. Eyben, S. Böck, B. Schuller, and A. Graves, "Universal onset detection with bidirectional long-short term memory neural networks," in *Proc. 11th Intern. Soc. for Music Information Retrieval Conference, ISMIR, Utrecht, The Netherlands*, 2010, pp. 589–594.

[25] B. Li, K. Dinesh, Z. Duan, and G. Sharma, "See and listen: Score-informed association of sound tracks to players in chamber music performance videos," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 2906–2910.

[26] B. Li, C. Xu, and Z. Duan, "Audiovisual source association for string ensembles through multi-modal vibrato analysis," *Proc. Sound and Music Computing (SMC)*, 2017.

[27] B. Li, K. Dinesh, C. Xu, G. Sharma, and Z. Duan, "Online audio-visual source association for chamber music performances," *Transactions of the International Society for Music Information Retrieval*, vol. 2, no. 1, 2019.

[28] B. Li, X. Liu, K. Dinesh, Z. Duan, and G. Sharma, "Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications," *IEEE Transactions on Multimedia*, vol. 21, no. 2, pp. 522–535, 2018.

[29] G. Bastas, A. Gkiokas, V. Katsouros, and P. Maragos, "Improving audio onset detection for string instruments by incorporating visual modality," *MML 2020*, p. 32.

[30] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[31] B. Li, A. Maezawa, and Z. Duan, "Skeleton plays piano: Online generation of pianist body movements from midi performance." in *ISMIR*, 2018, pp. 218–224.

[32] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[33] C. Lea, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks: A unified approach to action segmentation," in *European Conference on Computer Vision*. Springer, 2016, pp. 47–54.

[34] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[35] Visual onset detection from bowed strings. [Online]. Available: https://github.com/gbastas/avOnset.git

[36] J.-C. Hou, S.-S. Wang, Y.-H. Lai, Y. Tsao, H.-W. Chang, and H.-M. Wang, "Audio-visual speech enhancement using multimodal deep convolutional neural networks," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 117–128, 2018.

BEAT ESTIMATION FROM MUSICIAN VISUAL CUES

Sutirtha CHAKRABORTY (sutirtha.chakraborty@mu.ie) (0000-0001-8635-8860)[1],
Senem AKTAŞ (0000-0002-3996-2771)[2], **William CLIFFORD**[1], and **Joseph TIMONEY**[1]

[1]**Maynooth University**, Maynooth, Ireland
[2]**Bolu Abant İzzet Baysal University**, Bolu, Turkey

ABSTRACT

Musical performance is an expressive art form where musicians interact with each other using auditory and non-verbal information. This paper aims to discover a robust technique that can identify musical phases (beats) through visual cues derived from a musician's body movements captured through camera sensors.A multi-instrumental dataset was used to carry out a comparative study of two different approaches: (a) motiongram, and (b) pose-estimation, to detect phase from body sway. Decomposition and filtering algorithms were used to clean and fuse multiple signals. The final representations were analysed from which estimates of the beat, based on a 'trust factor', were obtained. The Motiongram and pose estimation were found to demonstrate usefulness depending on the musical instrument as some instrument playing gestures stimulate more movement in the players than others. Overall, the results were most promising using motiongram. It performed well where string instruments were used. The spatial derivative technique based on human pose estimation was consistent with woodwind instruments, where only a small degree of motion was observed.

1. INTRODUCTION

Human bodies typically move along to musical rhythms and studies conducted on infants have shown that already at such a young age there is a strong sensory link between body movement and auditory rhythm processing [1]. When people listen to musical performances, their mind subconsciously starts participating with it. Audience members can be seen to respond by tapping their feet, clapping their hands, or nodding their head. Musicians also move their body in different ways to perform expressive pieces depending upon the instruments they are playing. These movements enable them to express and portray their musical ideas [2], and their body movements an have a significant effect on musical rhythm perception. Understanding the underlying dynamics of this strong correlation between music and movement has been identified as an important component within the music, art, and engineering interdisciplinary field [3]. Non-verbal gestures and movements also provide musicians with a strongly connected communication between each member of the ensemble during performance as well as the audience. The interaction adds a unique and immediate dimension to a live performance that is unavailable from a recording. There is a constant transfer of information through the expressive movements of the musicians during the performance offering an integrated auditory and visual experience. Wanderley et. al, [4] studied the performances of five clarinettists to analyze the significance of gesture in three different ways: immobile, standard and expressive. The results of this experiment emphasized the presence of ancillary gestures which were consistent throughout the performance and were an integral part of musical performances. It is difficult to get musicians to perform completely still and change ingrained movements. Conscious and unconscious movements by musicians are made during the performance to make the performance look natural and expressive.

There exists a large computational complexity in correctly decoding musical cues from body movements using video data, and so it still remains a challenging task [5] [6]. Additionally, accuracy in identification is not obtained easily. In this paper, we focus on analysing the connection between musicians' body movements and the associated audio rhythms. We evaluated the performance of two different approaches to understand and predict rhythmic phases or beats from multiple musician's while they are performing. The method developed offers a general representation and approach that is independent of the number of instruments and musicians.

2. RELATED WORK

There are some examples in the literature of multi-modal methods of ascertaining the rhythm of a performance, that is, by not analysing the musical signal alone, and incorporating other information sources. As might be expected, the wide availability of video encourages the use of gestural cues but others have taken more novel perspectives. In the category of novel is a very recent study by [7] on Indian Classical Dance. Here, the music of the dancing combined with features extracted using speech processing techniques were analysed for rhythm marking.

Intuitively, it is expected that there should be a correlation between gestural behaviour and the rhythm of a performance. However, the connection is not fully understood. Some work has been put into furthering our understanding, others have looked at synthesizing gestural and audio

data. In the former case, preliminary findings have been obtained regarding the manifestation of period and phase-locking behaviour in full-body movements associated with how people participate in music. [8]. More comprehensive measurements of whole-body movement were also made. In the study of [2], the freedom of use of the head and upper body in the movements of pianists were examined along with their relationship to the performance. To better understand how to synthesize a convincing interaction between movement and sound, animation and video manipulation have been used. A method based on pixel motion sonification was introduced by [9]. Its success led to the analytical tool known as the Motiongram (to be discussed in Section 4). In the study [10], they did a similar manipulation of an animation. They used an LSTM network, trained using the input sound signal (piano and violin) along with measured gestures, to estimate the positions (termed as body keypoints) on an animated skeleton. This was the first time such a procedure was attempted using these keypoint features, and the demo was rated as working well. Similarly, Liu et al. [5] involved the recreation of violinists' body movements from music recordings and videos. Improvements and additions were made to the machine learning model to account for the finger positions and the orientation of the 3D skeleton. Kao and Su [3] generated a model formed in a U-net based encoder architecture, that included two additional network architectures created for the body and the right hand, and a self-attention mechanism. This was used to animate a virtual violinist's skeletal movements in 3D synchronized with a music signal. Moving away from animations, Davis and Agrawala [11] studied visual rhythm from videos, seeking associated features that are manifested in the temporal dynamics of the visible motion. They applied this to develop a novel way for manipulating actual performances in a dance video, keeping the movement and audio in synchronization. Lastly, in tandem with this thread in the research, other authors have focused on creating better beat tracking algorithms and analysis by including gestural cues. An audio-visual rhythm analysis was carried out by tracking body skeleton features detected using a Kinect sensor combined with the audio signals in [12]. They fused the video and audio data using a 'tempo likelihood' to create a predictor for the tempo and beat. It was reported that the video cues outperformed the traditional audio beat tracking in a noisy environment in the tempo prediction. To summarise, mixing gestural and audio data would improve beat and rhythm estimation. However, the method to do this is not fully established. This paper will contribute by developing algorithms that can work on features derived from representations of performances extracted from video. The next section introduces the dataset and the experimental setup. The following section goes through the methodology, explaining the Motiongram and pose estimation representations and the features extracted from them. There is a set of results presented along with a discussion. The paper ends with some conclusions and recommendations for future work.

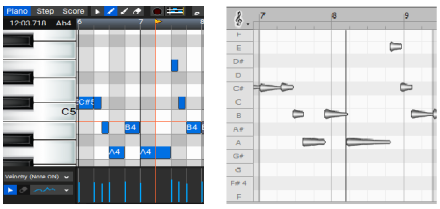


Figure 1. Example of the Midi and Audio representations (left and right panel respectively) used for their alignment to determine the beats

3. DATASET AND EXPERIMENTAL SETUP

We used an audio visual multi-instrument dataset, which contains 44 full musical performances [13]. The dataset is ideal for this experiment as the musical tracks had a significant tempo and clearly observable musical leader-follower exchanges unlike a typical live orchestra. As later in part 5, we evaluate on fugue composition where a short phrase is successively introduced from multiple musicians behaving as a 'lead', while others 'follow' the cue. The dataset contains four major types of performances: 11 Duets, 12 Trios, 14 Quartets, 7 Quintets.

For this study, three random videos were selected from each of the sections. The dataset also provided the audio, MIDI, and musical notations of each performance used to generate the ground truth.

3.1 Finding The Audio Beat

To understand musical phases, the time of the audio beats are important. The audio beats were considered as the ground truth value in our experiment. To compute audio beats, an audio-MIDI alignment technique was used [14]. As shown in Fig 1, the MIDI file representation (left hand side) was converted to its equivalent audio (right hand side). Dynamic Time Warping (DTW) was applied to effectively match and align the music recordings with the corresponding MIDI transcriptions. After which, the beats were estimated from the aligned-MIDI information. This process was repeated for the whole dataset.

4. METHODOLOGY

For estimating the musical phases of the video signal, firstly the video needed to be converted into a 1-D time-series signal. Two different approaches were used to identify a robust and scalable representation with respect to the number of musicians in the frame. In this experiment we used the following:

4.1 Video to Motion Signal

To convert the videos to a signal representing motion, we investigated two different approaches. They were :

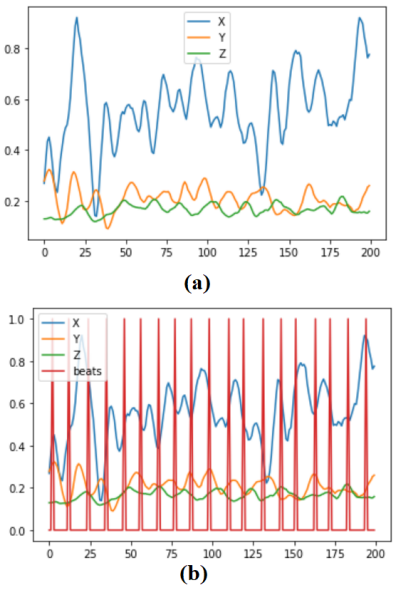


Figure 2. Three signals generated from motiongram data of a video (a), *X*: Horizontal motion ,*Y*: Vertical motion, *Z*: Quantity of motion (b) The three *X*, *Y* and *Z* motion signals with the time instances of the audio beat superimposed on them

4.1.1 Motiongram

The motiongram was created by calculating the spatio-temporal motion using the average pixel intensity change between frames of video. It provides a series of images showing motion in two dimensions across the image plane, I with dimensions $n \times m$, for some small time-step dt, see Equation 1. The signals generated with respect to the motiongram are shown in Fig 2. In the plot in the upper panel the *X*-value refers to the average horizontal pixel positions in the image, while the *Y*-value refers to the average vertical pixel positions. The other feature used was the 'quantity of motion' [15], that represented the overall average amount of motion at that time-step. The signals were normalized independently for further processing. The lower panel shows the relationship between the *X*, *Y* and *Z* signals and the locations of the beats in the music. Change in number of participants was found to not effect the output of motiongram.

$$I(t+dt) = |I(x,y,t+dt) - I(x,y,t)| \in \mathbb{R}^{n \times m} \quad (1)$$

4.1.2 Pose Estimation

Recognizing human activities from video sequences is a challenging task. While humans can do this easily, it is a more complex process for computers to recognize human activities efficiently, and such operations are based on multimodal activity recognition methods [16]. The pose estimation algorithm extracted the 2D coordinates of human body joint locations (such as arms,head) from videos.

To use pose estimation, we considered each musician as an individual oscillator. The average body movements generated by each of them was interpreted as a 1D motion signal. We used openpose, an improved and robust model pose estimation model that applies Part Affinity Fields (PAFs) to predict body keypoints for multiple humans [17]. These coordinates were used to compute the full body motion for each musician. It generated, 17 body keypoints, $\mathcal{K}$, from each frame of the video. The inter-keypoint distance of each individual was recorded over time from their body-center.

Two different approaches were tested with pose estimation. They are :

- First Frame as Reference
- Spatial Derivative of keypoints

In the first technique, the first frame was considered as the point of reference for calculating the motion. An average value was computed to find the relative motion resulting in a 1D signal for each musician as shown in the lower panel of Fig 3. This measured the variances of the keypoints for each time-step, see Equation 2 [1] where $k \in \mathbb{R}^2$ is each keypoint, $\bar{k}$ is the keypoint average, and n is the number of keypoints $\in \mathcal{K}$. Each musician was considered as an individual signal source. Therefore, with the increase in number of participants, there were more signals present.

$$\sigma_t = \frac{\sum_{i=1}^{n} L_2(k_i, \bar{k})}{n} \in \mathbb{R} \quad (2)$$

The spatial derivative was used in the second technique to generate the average motion signal of each individual by comparing each frame's keypoints $\mathcal{K}_t$ with its previous frame, similar to optical flow [18]. Given that the points had already been localized, there was no need to calculate the intensity-based image gradients. The spatial derivative, $\Delta \boldsymbol{k}$, was calculated for each keypoint $\in \mathcal{K}_t$ using Equation 3, where k is each keypoint on the image plane. To get a measurement of how much motion was associated with each keypoint at time-step dt, the magnitude of the vector $\Delta \boldsymbol{k}$ was calculated, and an overall average μ_t for all the keypoints $\Delta \boldsymbol{k}_i$ at that time-step n was found, as given in Equation 4.

$$\Delta \boldsymbol{k} = k(t+dt) - k(t) \in \mathbb{R}^2 \quad (3)$$

$$\mu_t(\Delta \mathcal{K}) = \frac{\sum_{i=1}^{n} \|\Delta \boldsymbol{k}_i\|}{n} \in \mathbb{R} \quad (4)$$

4.2 Post Processing

4.2.1 Filtering the signal

The signals generated by motiongram and pose estimator showed a rhythmic pattern when they were plotted. The Okada filter was used for smoothing the dataset [19]. This was used as it can increase the data precision without distorting or affecting the position of the signal peak. It is

[1] $L_2(\cdot,\cdot)$ represents the L_2 norm between two points or the Euclidean distance

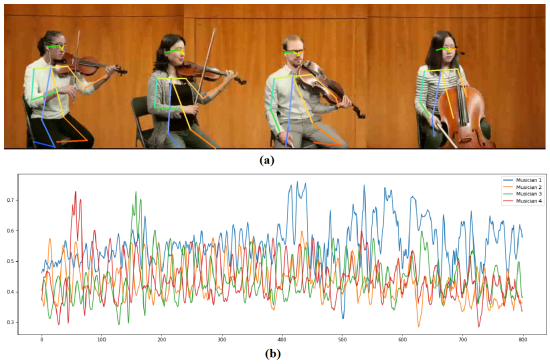


Figure 3. The upper panel shows a frame of the video of the musicians where each musician represents as an oscillator and the lower frame gives the average motion calculated by pose estimation on each musician based on the first frame as reference and the spatial derivative of keypoints from each participants

based on logistic function over a single continuous differentiable equation as shown in Equation 5. Here, x_t refers signal value(x) at time t and x_{t-1} and x_{t+1} are at the previous and next time points respectively. α is a weighting factor and is typically set to be 100.

$$x_t \leftarrow x_t + \frac{x_{t-1} + x_{t+1} - 2x_t}{2(1 + e^{-\alpha(x_t - x_{t-1})(x_t - x_{t+1})})} \quad (5)$$

4.2.2 *Matrix Decomposition*

We considered each signal as a dimension.Thus, using a matrix decomposition algorithm,we converted multiple signals into a single dimension. This one dimensional signal showed a high correlation with the audio beats (ground truth). We investigated five different dimensional reduction algorithms [20]. These included :

- Incremental principal components analysis (IPCA)
- Principal component analysis (PCA)
- Kernel Principal component analysis (KPCA)
- Dimensionality reduction using truncated SVD (aka LSA)
- FastICA: a fast algorithm for Independent Component Analysis

The multi-collinearity was handled by removing the redundant features. In Fig 4, the five dimension reduced signals were used for both motiongram and pose estimation output. The motiongrams horizontal(x-value), vertical image plane values(y-value) and quantity of motion(QOM-Value) were decomposed together, as shown in Fig 4(a). However,during these performances, there were multiple musicians interacting with each other.The body keypoints of each musician was considered as an individual independent oscillator generating a signal. For example, in a quartet performance,four average motion signals were treated

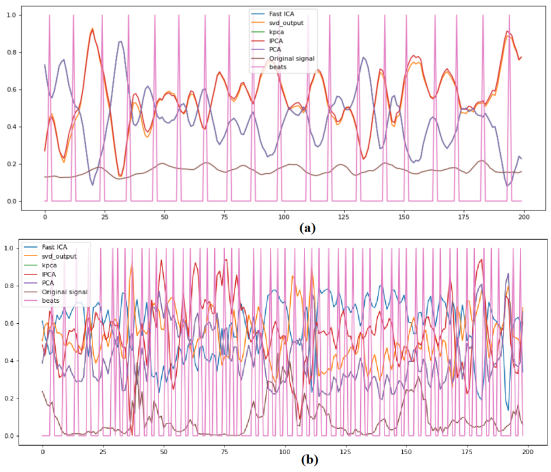


Figure 4. 1D signals generated after dimensional reduction of the motion signals with five different algorithms(a) Three dimensions(X,Y,QoM) of motiongram were reduced to one-dimensional signal (b)Reduced one-dimensional signals obtained from multiple musicians' pose estimated motion

as four different dimensions. Additionally, matrix factorization techniques were used to combine the movement data as a single oscillator for phase prediction.

4.2.3 *Peak and Valley Detection*

Each decomposed signal provided its own set of timestamps. A peak detection algorithm [21] was used to detect the peaks and the valleys for each of the five decomposed signals. The peaks and valleys were then merged and sorted in increasing order of time. The times of all the five signals were superimposed on each other. A voting-based system was used for selecting phases. Any three of the signals providing matching timestamps at their peaks/valleys were considered (following cut off thresholds). Other peaks/valleys were ignored. Strong points were considered to be visual beats and compared with the audio beats (ground truth) as shown in Fig 5.

5. RESULTS

For the evaluation [2] we selected three from each of the four types of performance, as shown in Table 1 which had not been used for the training.The classical pieces were composed by renowned composers including Holst, Mozart, Joplin, Bach and others. The preference for classical music is that it is often performed with rubato and thus its tempo more elastic than the beat-driven pop or rock music. In addition to this, beat detection can be difficult for instruments of the string, brass, and wind family as their onsets are less well-defined when not played in a percussive style. Overall,this detection problem is exacerbated by the absence of drums in most classical music.The musical pieces were selected in such a way so that all types of musical instru-

[2] https://github.com/SutirthaChakraborty/motiongramVsPoseEstimation

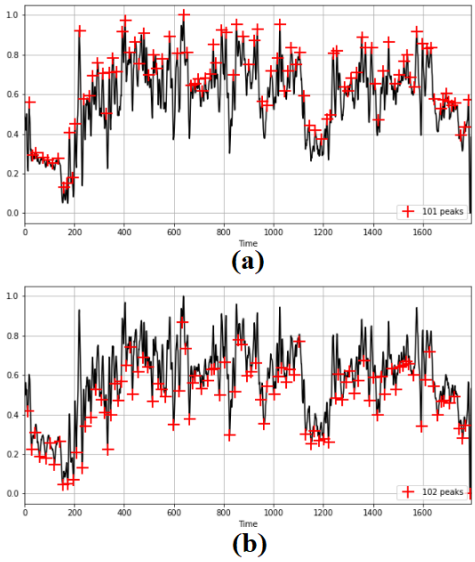


Figure 5. To calculate the phase of the movements,(a) Peaks and (b) Valleys were detected from dimensional reduced signals and are shown by red crosses in the upper and lower panels respectively.

ment were included in the experiment, and thus we could precisely identify the impact of the instruments' family on the accuracy of the two different approaches used for the video to signal conversion. Musicians can often exhibit an instrument-specific style of movement that helps them to maintain the rhythm while playing [22].For computational accuracy when making the evaluation, we considered two cases: Firstly, that the beats that were exactly predicted by the algorithms and secondly, that the beats that lay within a bound of 100ms on either side of the ground truth beat's timestamp. The value of 100ms was chosen as since the videos are 29.75 frames per second (FPS), it means within three video frames either side of the audio beat event there must be a corresponding video beat event [23]. On executing the evaluation, we found that the best detection approach depends mostly on the types of instrument played and on the number of musicians playing, as shown in Fig 6. During validation, we achieved a maximum F1-score of 0.34 when using the motiongram, followed by a score of 0.33 when using optical flow to predict the audio beats of each performance. The First Frame Reference approach had the worst result of 0.32. The full validation results are shown in table 2. In Fig 6 the Motiongram output values (shown by the blue bars) were found to be most distinct for string instruments. This was expected as the musicians exhibit greater levels of activity while playing. In contrast, the pose estimation with the optical flow method worked well with woodwind instruments as those musicians display minimal motion. Considering all the instrument families, the Motiongram was found to be more reliable overall and yielded a more consistent performance with an increase in the number of musician participants. We also

Type of Performance	Name	Duration	Instrument*	Total Beats
Duet	Jupiter	01:03	Vn,Vc	86
	Sonata	00:46	Vn,Vn	44
	The Entertainer	01:27	Tpt, Tpt	216
Trio	Spring for the Four Seasons	00:35	Vn,Vn,Vc	65
	Hark the Herald Angels	00:47	Vn,Vn,Va	88
	Waltz from Sleeping Beauty	01:33	Fl,Fl,Cl	304
Quartets	Pirates of the Aegean	00:50	Vn,Vn,Va,Vc	163
	Pirates of the Aegean	00:50	Vn,Vn,Va,Sax	163
	In the Hall of Mountain King	01:25	Vn,Vn,Va,Vc	160
Quintets	Miserere Mei Deus	00:40	Fl,Fl,Ob,Cl,Bn	87
	Miserere Mei Deus	00:40	Fl,Fl,Ob,Cl,Bn	86
	Chorale	00:53	Tpt,Tpt,Hn,Tba,Tba	144
*Instruments : Vn=Violin, Vc=Cello, Tpt= Trumpet, Va= Viola, Fl=Flute, Ob=Oboe, Cl=Clarinet, Bn=Basson, Tba= Tuba				

Table 1. Performances selected for evaluation

	Motiongram	First Frame Reference	Optical Flow
F1-Score	0.34	0.32	0.33
Precision	0.35	0.34	0.33
Recall	0.32	0.30	0.31

Table 2. Validation Score

found that musicians frequently tend to move their body in anticipation of the beat [24]. While not performing as well as the Motiongram overall the optical flow approach was reasonably consistent in its output, as can be seen in the figure. The pose estimation approach, where we considered the first frame as the reference frame, did not make reliable predictions as observed by the varying nature of the values plotted in Fig 6.

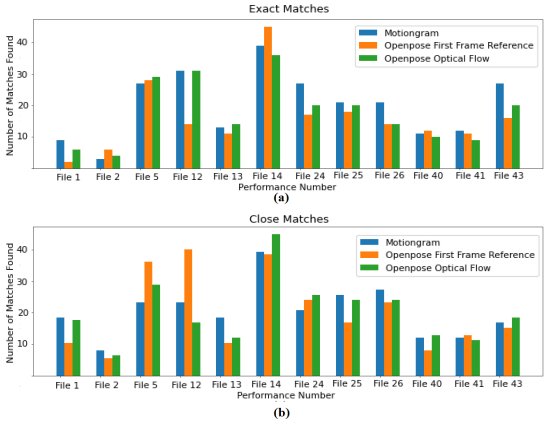


Figure 6. The evaluation procedure was carried out on 12 performances to find Fig (a): Quantity of Exact Matches: the ground truth audio phase matched with predicted phase from videos Fig (b): Quantity of Close matches:acceptable lag of the audio phase that is 100ms second apart from the predicted video phase.

6. CONCLUSION

This study shows how rhythmic phases could be predicted from a music ensemble or group performance using only video cues. Although difficult for the human eye to ascertain consistently, this approach could identify poten-

tially consistent features from the non-verbal communication that exists among musicians and how their body and gestural movements are engaged in maintaining the phase synchronization across beats during a musical performance. The study considered how the physical playing activity could be considered as oscillators and how this could characterise the underlying rhythmic dynamics of their body movements. We presented a promising set of techniques for predicting the beat locations utilizing data derived from a video recordings of the musicians' bodies. We can conclude from our results that the approach is influenced heavily by the instruments that are being played. If the instruments allow or encourage more movement during playing, such as string instruments, motiongrams and pose estimation combined with an optical flow technique would be more appropriate for predicting the musical rhythm. Pose estimation alone was more effective with instruments that inspire less body movement, such as those of the woodwind family.

For future work, the most immediate concern is to improve the accuracy by developing a post-processing technique that will eliminate sources of error that lead to false positives. Another idea is to develop a multimodal approach that fuses the predictions derived from the audio and video data. Also, at a higher level it would be useful to have an automated decision-making system that would simply choose the most appropriate analysis technique by identifying the music instrument type from the audio signal. In the longer term, video footage of larger ensembles of instruments will be studied, more data will be collected, and a greater use of machine learning technologies will be made.

7. REFERENCES

[1] J. Phillips-Silver and L. J. Trainor, "Feeling the beat: movement influences infant rhythm perception," *Science*, vol. 308, no. 5727, pp. 1430–1430, 2005.

[2] M. R. Thompson and G. Luck, "Exploring relationships between pianists' body movements, their expressive intentions, and structural elements of the music," *Musicae Scientiae*, vol. 16, no. 1, pp. 19–40, 2012.

[3] H.-K. Kao and L. Su, "Temporally guided music-to-body-movement generation," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 147–155.

[4] M. M. Wanderley, B. W. Vines, N. Middleton, C. McKay, and W. Hatch, "The musical significance of clarinetists' ancillary gestures: An exploration of the field," *Journal of New Music Research*, vol. 34, no. 1, pp. 97–113, 2005.

[5] J.-W. Liu, H.-Y. Lin, Y.-F. Huang, H.-K. Kao, and L. Su, "Body movement generation for expressive violin performance applying neural networks," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 3787–3791.

[6] F. Desmet, M. Leman, M. Lesaffre, and L. De Bruyn, "Statistical analysis of human body movement and group interactions in response to music," in *Advances in data analysis, data handling and business intelligence*. Springer, 2009, pp. 399–408.

[7] T. Mallick, P. P. Das, and A. K. Majumdar, "Beat detection and automatic annotation of the music of bharatanatyam dance using speech recognition techniques," *arXiv preprint arXiv:2004.08269*, 2020.

[8] B. Burger, M. R. Thompson, G. Luck, S. H. Saarikallio, and P. Toiviainen, "Hunting for the beat in the body: on period and phase locking in music-induced movement," *Frontiers in human neuroscience*, vol. 8, p. 903, 2014.

[9] A. R. Jensenius, "Motion-sound interaction using sonification based on motiongrams," in *Proceedings of the ACHI 2012: The Fifth International Conference on Advances in Computer-Human Interactions.*, 2012, pp. 170–175.

[10] E. Shlizerman, L. Dery, H. Schoen, and I. Kemelmacher-Shlizerman, "Audio to body dynamics," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7574–7583.

[11] A. Davis and M. Agrawala, "Visual rhythm and beat," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 2532–2535.

[12] M. Ohkita, Y. Bando, E. Nakamura, K. Itoyama, and K. Yoshii, "Audio-visual beat tracking based on a state-space model for a robot dancer performing with a human dancer," *Journal of Robotics and Mechatronics*, vol. 29, no. 1, pp. 125–136, 2017.

[13] B. Li, X. Liu, K. Dinesh, Z. Duan, and G. Sharma, "Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications," *IEEE Transactions on Multimedia*, vol. 21, no. 2, pp. 522–535, 2018.

[14] C. Raffel and D. P. Ellis, "Optimizing dtw-based audio-to-midi alignment and matching," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 81–85.

[15] A. R. Jensenius, "The musical gestures toolbox for matlab," in *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*. Institut de Recherche et Coordination Acoustique/Musique, 2018.

[16] M. Vrigkas, C. Nikou, and I. A. Kakadiaris, "A review of human activity recognition methods," *Frontiers in Robotics and AI*, vol. 2, p. 28, 2015.

[17] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[18] D. Fleet and Y. Weiss, "Optical flow estimation," in *Handbook of mathematical models in computer vision*. Springer, 2006, pp. 237–257.

[19] M. Okada, T. Ishikawa, and Y. Ikegaya, "A computationally efficient filter for reducing shot noise in low s/n data," *PLoS One*, vol. 11, no. 6, p. e0157595, 2016.

[20] S. Rahaman, S. Shahabuddin, M. B. Hossain, and S. Shahabuddin, "Complexity analysis of matrix decomposition algorithms for linear mimo detection," in *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*. IEEE, 2016, pp. 927–932.

[21] "Sensor Motion python package for analyzing sensor-collected human motion data (e.g. physical activity levels, gait dynamics)," https://github.com/sho-87/sensormotion.

[22] C. Palmer, C. Carter, E. Koopmans, and J. D. Loehr, "Movement, planning, and music: Motion coordinates of skilled performance," in *Proceedings of the International Conference on Music Communication Science*. University of New South Wales Sydney, NSW, 2007, pp. 119–122.

[23] R. H. Jack, A. Mehrabi, T. Stockman, and A. McPherson, "Action-sound latency and the perceived quality of digital musical instruments: Comparing professional percussionists and amateur musicians," *Music Perception: An Interdisciplinary Journal*, vol. 36, no. 1, pp. 109–128, 2018.

[24] A. D. Patel and J. R. Iversen, "The evolutionary neuroscience of musical beat perception: the action simulation for auditory prediction (asap) hypothesis," *Frontiers in systems neuroscience*, vol. 8, p. 57, 2014.

EXPECTED RECIPROCAL RANK FOR EVALUATING MUSICAL FINGERING ADVICE

David A. RANDOLPH (drando2@uic.edu)[1], **Barbara DI EUGENIO**[1], and **Justin BADGEROW**[2]

[1]*Department of Computer Science*, **University of Illinois at Chicago**, IL USA
[2]*Department of Music*, **Elizabethtown College**, PA USA

ABSTRACT

We cast the computational modeling of musical fingering as an information retrieval (IR) problem in which the task is to generate an optimally ranked list of fingering suggestions for each phrase in a score. The audience for this list is a set of performers with potentially diverse fingering preferences. Specifically, we adapt the expected reciprocal rank (ERR) metric—proposed by Chapelle and associates as an improved evaluation metric for retrieving documents with graded relevance—to develop a set of novel metrics tailored to the piano fingering IR task. ERR, as originally described, relies on a heuristic function to estimate the probability that a user will be satisfied by a document with a particular graded relevance. For musical fingering, we instead estimate the likelihood that a given performer will deem a suggested fingering sequence sufficient for arriving at a satisfactory solution. Finally, we attempt to validate our specific use of ERR by comparing how it judges several competing models.

1. INTRODUCTION

Pianists typically encounter fingering advice as annotations on a static printed score. In some cases, especially for more difficult repertoire, pianists may own more than one editorial score to obtain a variety of fingering suggestions. A key advantage of an automated advice generation system, therefore, would be its ability to provide a variety of advice on demand. It is also a common feature of existing models to output ranked lists of fingering sequences.

Moreover, assessment methods for the published piano fingering models are inadequate. With few exceptions [1–3], the extrinsic evaluations performed on proposed models consider only a single authoritative source of "correct fingerings." The same can be said of how models of guitar fingering have been evaluated. This is a missed opportunity, as is affirmed by the variability in the domain described qualitatively by [1] and [4]. The existence of multiple ground truths must be acknowledged and accommodated in model evaluation.

Parncutt et al. [1] collect "preferred fingerings" from a set of 28 pianists with unreported hand dimensions or gender, ranking particular fingerings by how many pianists prefer them. Their computer model is deemed a success because, "In most cases, the most popular fingering [among pianists] is in the top 10 [selected by the model]." Jacobs [2] uses an identical evaluation technique, and boasts of an improvement because "more pianist's fingerings are now included in the top 10." Nakamura et al. [3] confront the problem squarely and suggest several methods for comparing automatically generated advice with single or multiple ground truths. These methods rely on perfect matches at each note position and contemplate evaluating only one automatically generated fingering sequence at a time. They also describe a "recombination match rate," which involves calculating a cost of "constructing" a generated fingering sequence by combining the ground truths that are available.

Our approach here is more straightforward and includes multiple system outputs as part of the evaluation framework, as seems appropriate for a domain acknowledged to contain multiple ground truths and for automated systems that should be expected to produce diverse fingering suggestions.

We therefore cast the development of fingering models as an information retrieval (IR) task. Given arbitrary musical input (a "query"), the system generates a list of the most relevant fingerings ("documents"), ordered optimally to satisfy the information need of the pianist ("user.") With this framing of the task, we draw on recent advancements in evaluation measures for IR systems to develop a set of novel evaluation metrics for piano fingering.

Here we demonstrate an application of ERR to piano fingering advice, but the general approach should be appplicable to any instrument where fingering decisions form an important part of skilled performance. At various times, accordianists, string players, and many percussionists may also be eager consumers of fingering (or sticking) advice. Other instruments (e.g., brasses and woodwinds) typically have few fingering choices, making computational models less relevant for them.

To simplify explication, in our application of ERR to the evaluation of piano fingering advice, we reduce the problem space to include only monophonic (melodic) musical phrases played with the right hand.

Open-source implementations of all methods described below are released as part of Pydactyl [5] at `https://github.com/dvdrndlph/pydactyl`. Full release of the corpora used in validation is forthcoming.

2. EXPECTED RECIPROCAL RANK

Specifically, we adapt expected reciprocal rank (ERR), proposed by [6] as an improved metric for search engines when retrieving documents with graded relevance, to the musical fingering IR task. ERR generalizes the Reciprocal Rank (RR) metric of [7], a simple assessment for documents with binary relevance: the quality of a list of documents in which the first relevant document appears at rank r is estimated as $\frac{1}{r}$. To calculate ERR(R) for a returned list of R documents, system evaluators define a function to estimate the probability that a user will be satisfied by a document with a certain graded relevance. Armed with these probabilities, it is then straightforward to determine the likelihood that a user will be satisfied after reviewing a document at each rank. These likelihoods, after being assessed a reciprocal-rank $\frac{1}{r}$ (or similar) discount, are summed to calculate ERR(R) for a list of length R. The basic intuition of ERR is summed up in Equation 1, as defined by [6]:

$$\text{ERR}(R) = \sum_{r=1}^{R} \frac{1}{r} P(\text{user stops at position } r). \quad (1)$$

The details for leveraging the individual probability estimates P_i (that each recommended document in the list will satisfy the user) to determine the likelihood that a user stops the search at rank r are conveyed in Equation 2 (with notation slightly altered from the original). The key point here is that the probability of stopping at a rank r is the probability imputed from the graded relevance of the document at this rank *reduced by the probability that this rank is never reached by the user*—that is, that a document ranked higher has already satisfied the information need. Through this property, the ERR measure reflects a "cascade user model," which has been applied effectively to explain user behavior when interacting with web search results [8]:

$$\text{ERR}(R) = \sum_{r=1}^{R} \frac{1}{r} \prod_{i=1}^{r-1} (1 - P_i) P_r. \quad (2)$$

Chapelle and associates illustrate their metric with a probability estimation function for documents assigned an integer relevance score $g \in \{0, \ldots, g_{max}\}$:

$$\mathcal{P}(g_r) = \frac{2^{g_r} - 1}{2^{g_{max}}}. \quad (3)$$

Thus, the graded relevance of g_r of a document at rank r, the prior probability of the acceptability of document r may be estimated:

$$P_r = \mathcal{P}(g_r). \quad (4)$$

For the musical fingering problem, we adapt the ERR metric to entail distinct methods for estimating the prior probability that a document—that is, a fingering sequence—will be acceptable to the user. Instead of relevance grading, we propose evaluating the quality of a suggestion according to its similarity to a gold-standard fingering sequence.

We assume that any acceptable system advice, in the vast majority of cases, will be *similar* to satisfactory fingering sequences developed by the performer independent of suggestions from third parties or, more generally, to some acceptable advice suggested by more expert pianists. By necessity, fingerings developed by humans form the basis of the expertise being modeled. Crucially, we do not expect performers to require each note's suggested fingering to be identical to the one they ultimately adopt for that note.

The simplest similarity measure between two fingering sequences for a phrase is Hamming distance. In this measure, we simply count the number of individual elements in a sequence that do not match exactly: The fewer mismatches, the more similar the strings. Insofar as previously published fingering models have discussed performance of their models, they appear to refer to "accuracy" as the percentage of exact matches at the level of individual fingerings. This measure makes no assumptions about the process underlying how fingering decisions are made. Normalizing for the length N of the phrase, we have the following grading function g for the fingering at rank r produced by system S:

$$g_r = \frac{\Delta(H, S_r)}{N}, \quad (5)$$

where H is the fingering sequence ultimately adopted by the human pianist (and included in a gold-standard corpus) and Δ is the Hamming edit distance function:

$$\Delta(A, X) = \sum_{n=1}^{N} \delta(A_n, X_n), \quad (6)$$

where

$$\delta(a, x) = \begin{cases} 0 & \text{if } a = x, \\ 1 & \text{otherwise.} \end{cases} \quad (7)$$

Normalizing the Hamming distance conveniently implies $0 < g_r < 1$, so we may use the following to estimate the probability $\mathcal{P}$ that advice at rank r will satisfy the user:

$$\mathcal{P}_r = 1 - g_r. \quad (8)$$

2.1 Modified Unigram Edit Distance

Motivated by the observation that not all fingering deviations are equally significant, however, we pose a modification to simple Hamming distance for use in the piano domain. Clearly, choosing the index finger (2) over the middle finger (3) in many cases might seem arbitrary to the player, or at the very least a substitution readily made as circumstances allow. As such, a 2-3 or 3-2 variation is less likely to damage the overall opinion one holds of a complete sequence. This intuition is supported by pedagogues who place fingers 3 and 2 on near parity. Per [9, p. 115], "The third finger is by nature the skilfullest and strongest. The style of touch which it possesses serves for a time as a standard for the other fingers. . . . The second is the next strongest and skilfullest. Its mobility is probably greater, but in strength it yields to the third."

Selecting the middle finger (3) over the ring finger (4), though perhaps more controversial, is still a relatively minor deviation when compared to substituting more remote

	1	**2**	**3**	**4**	**5**
1	0	1	1	1	1
2	1	0	$\frac{1}{2}$	1	1
3	1	$\frac{1}{2}$	0	$\frac{1}{2}$	1
4	1	1	$\frac{1}{2}$	0	1
5	1	1	1	1	0

Table 1. Confusion matrix for one-handed "adjacent long" weighted edit distance for piano.

fingers. Fingers 2, 3, and 4 are the longest fingers and as such they are adept at playing both black and white keys. While 3 is stronger than 4 typically, these fingers are used interchangeably frequently in certain patterns and reflecting different pianists' approach to standardized structures.

We therefore propose an alternative weighted edit distance, defined as a confusion matrix in Table 1, as a more appropriate $\delta(a, x)$ for piano. We refer to the Δ function applying this confusion matrix as the "adjacent long" edit distance function.

2.2 Trigram Edit Distance

The edit distance functions described above measure deviations between fingerings of individual notes. We have reservations about applying such unigram measures to a problem that is fundamentally about transitions between fingered positions (at least for keyboard and string instruments) and also about planning for future transitions. We therefore here pose alternatives, which measure consistency in two fingering sequences within a sliding window of three-note groups—τ trigram distance functions. The simplest (and most unforgiving) of such measures, described formally in Equation 9, requires exact matches for the finger used to play each note in the group:

$$\tau(A, X, n) = \begin{cases} 1 & \text{if eq}(A^n_{n-2}, X^n_{n-2}), \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

where

$$\text{eq}(abc, xyz) = \begin{cases} 0 & \text{if } a \neq x \text{ or } c \neq z \text{ or } b \neq y, \\ 1 & \text{otherwise.} \end{cases} \tag{10}$$

The added context of the trigram allows us, for piano, to incorporate the concept of adjacent long finger similarity discussed above with more precision. In our "nuanced" formulation, captured in Equation 11, we discount mismatches involving adjacent long fingers by some ε only for the middle note in a trigram and only when the surrounding notes are fingered identically. That is,

$$\tau(A, X, n) = \begin{cases} 0 & \text{if eq}(A^n_{n-2}, X^n_{n-2}), \\ 1-\varepsilon & \text{if sim}(A, X, n), \\ 1 & \text{otherwise,} \end{cases} \tag{11}$$

given equal(abc, xyz) is defined as above in Equation 10,

$$\text{sim}(A, X, n) = \begin{cases} 1 & \text{if eq}(A^n_{n-2}, X^n_{n-2}), \\ 0 & \text{if } A_{n-2} \neq X_{n-2} \text{ or } A_n \neq X_n, \\ 1 & \text{if proxy}(A_{n-1}, X_{n-1}), \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

and where

$$\text{proxy}(a, x) = \begin{cases} 1 & \text{if } a \in \{2, 3\} \text{ and } x \in \{2, 3\}, \\ 1 & \text{if } a \in \{3, 4\} \text{ and } x \in \{3, 4\}, \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

with $0 \leq \varepsilon \leq 1$. Clearly, the decision made for the middle note in a trigram has essentially no bearing on the fingerings outside the scope of the trigram when the outer two notes are fingered identically. A pianist should be quite forgiving of such deviations and be able to substitute their own preferences easily.

We suspect that pianists will be similarly accepting of these similarities, bracketed as they always are by fingerings with which they completely agree, when they appear in other positions (first or third) in other trigrams. We therefore pose one final "relaxed" τ distance function:

$$\tau(A, X, n) = \begin{cases} 0 & \text{if equal}(A^n_{n-2}, X^n_{n-2}), \\ 1-\varepsilon & \text{if simat}(A, X, n-2) \text{ and} \\ & \quad \text{simat}(A, X, n-1) \text{ and} \\ & \quad \text{simat}(A, X, n), \\ 1 & \text{otherwise,} \end{cases} \tag{14}$$

where

$$\text{simat}(A, X, m) = \begin{cases} 1 & \text{if } A_m = X_m \text{ or} \\ & \quad \text{sim}(A, X, m+1), \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Note that any of these τ functions may be used in place of the δ functions included in Equation 5, like so:

$$\Delta(A, X) = \sum_{n=1}^{N} \tau(A^n_{n-2}, X^n_{n-2}). \tag{16}$$

Note that each note is evaluated in its full trigram context. That is, each individual note contributes to three trigram evaluations, with the first and last two evaluations including referencing positions outside the scope of the note sequence. Such blank or null values are considered equal when compared. To check everything completely, N must be increased by two for all calculations involving trigram measures. Table 2 demonstrates how the summation in Equation 16 aggregates for each iteration (1–9) over the seven-note phrase using the three candidate trigram functions. Here we assume $\varepsilon = 1$, but using a number slightly less than one (such as 0.99, as we do in the experiments described below) affords better differentiation.

3. GENERAL PROBABILITY FORMULATION

We apply this measures into our estimate for the probability that system S advice at position r in a ranked list will

n	**-1**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
H	␣	␣	2	5	3	5	2	3	1	␣	␣
S_r	␣	␣	3	5	4	5	3	4	2	␣	␣
Match	$=$	$=$	$\approx$	$=$	$\approx$	$=$	$\neq$	$\neq$	$\neq$	$=$	$=$
τ_{trigram}			1	2	3	4	5	6	7	8	9
τ_{nuanced}			1	1	2	2	3	4	5	6	7
τ_{relaxed}			0	0	0	0	1	2	3	4	5

Table 2. The calculation of $\Delta(H, S_r)$ using the competing τ trigram distance functions ($\varepsilon = 1$).

satisfy the human who prefers fingering sequence H like so:

$$\mathcal{P}_r = 1 - \frac{\Delta(H, S_r)}{N}. \tag{17}$$

We attempt to validate the utility of Equation 17 with various Δ definitions in the experiments described below.

4. EVALUATING THE EVALUATIONS

Chapelle et al. [6], as a preamble to the evaluation of their ERR metric applied to web search: "The evaluation of new metrics is challenging because there is no ground truth to compare with. Because of that, most papers that propose new metrics do not have direct evaluations." They proceed to defend the application of ERR to web search by leveraging click-through data as a surrogate for what they would ideally have—namely, actual people interacting with lists comprised only of documents with known grades. We find ourselves in a similar position here, as we do not have data from a set of pianists interacting with fingering advice from competing models. What we have are passages fingered by pianists and several model implementations.

4.1 Corpora

Our initial validation efforts here are performed using the fragments from Czerny's challenging *160 Kurze Übungen* [10] that form the basis for evaluation of the original piano fingering model [1]. Parncutt and associates [1] publish fingering data for the opening notes of seven exercises, ranging from four to eight notes in length. Of the 28 pianists who provided fingerings, 25 "performed regularly on a regular basis" and three were enrolled in an "undergraduate music program." They report a "mean total number of years practicing and/or performing = 31" for participants. Data from this cohort constitute what we call the "published" corpus.

In addition, using these same Czerny exercises, we have assembled a more extensive corpus via an online survey and a web application [11] built for the purpose. Subjects were recruited from personal acquaintances of the researchers, email lists published by several music teachers' associations (California, Florida, Georgia, Massachusetts, New Jersey, Ohio, Pennsylvania, and the greater Chicago metropolitan area), and music departments at universities and colleges across the United States and Canada. In all, 5345 recruitment emails were sent, asking recipients to complete the survey and/or forward it to potentially interested students and colleagues. From this, 352 people (6.6%) responded. For the subset of 199 subjects who provided enough information, we determine a median of 36 "years of piano study." Included in this *full* corpus are all complete fingering sequences provided by any participant.

We deem all these data to represent expert fingering advice. Details about each corpus are provided in Table 3.

4.2 Models

To put ERR and our competing probability estimation methods through their paces, we leverage our own implementations [5] of the original piano fingering model described by Parncutt et al. [1] (hereafter referred to as the *parncutt* model) and the enhancement of this model described by Jacobs [2] (the *jacobs* model).

We use the distributions of the *published* and *full* corpora to synthesize two high quality models, which we dub *ideal-p* and *ideal-f*, respectively. The "system" lists produced by such models are composed of the most popular fingering suggestions from a large group of expert pianists. An automated system this good would surely define the state of the art.

Finally, we define two models—*random-p* and *random-f*—that simply apply a random fingering (1–5) to every note in a sequence, generating intentionally sub-optimal ranked "system" lists. These models should clearly underperform any reasonable advice generating system. (We seed the random number generator with a constant arbitrary value to guarantee reproducible results.)

4.3 Approach

For each corpus, we apply four models to each piece to obtain 28 S ranked fingering lists of length five. We then calculate five different ERR scores for each H fingering provided by a human, applying a different Δ function to estimate the probability P_r that the suggestion presented at rank r in the S list will satisfy the pianist. Each of the five estimates is determined by a different method: two that consider unigrams—Hamming distance (as in Equation 7) and "AdjLong" (as captured in Table 1)—and three that compare trigrams (either requiring exact trigram matches (Equation 9), nuanced matches (Equation 11), or yet more relaxed matches (Equation 14).

To summarize our results, we simply average the ERR scores achieved by each method to obtain a mean ERR value,

$$\text{MERR(A)} = \frac{1}{A}\sum_{i=1}^{A} \text{ERR}(H_i), \tag{18}$$

Corpus		*full*			*published*	
Piece	**Notes**	**Annotators**	**Annotations**	**Notes**	**Annotators**	**Annotations**
A (Op. 821 no. 1)	16	202	3232	8	28	224
B (Op. 821 no. 37)	16	201	3216	4	28	112
C (Op. 821 no. 38)	18	198	3564	5	28	140
D (Op. 821 no. 54)	16	190	3040	7	28	196
E (Op. 821 no. 62)	15	195	2925	8	28	224
F (Op. 821 no. 66)	16	192	3072	6	28	168
G (Op. 821 no. 96)	18	195	3510	7	28	196
Totals	115		22559	45		1260

Table 3. Details on Czerny corpora used in evaluation.

where A is the number of human-annotated phrases H in the corpus.

We expect to see ERR methods rank the four models in order from best to worst:

1. *ideal*
2. *jacobs*
3. *parncutt*
4. *random*

We also expect to see an advantage of trigram methods over unigram methods.

We use the *scipy* and *statsmosdels* python packages for statistical analysis.

4.4 Results

The MERR results are displayed in Table 4. The expected ordering of models is present for all four methods employed.

Eight one-way between-subjects ANOVAs conducted to analyze the differences reported between ERR means found all of them to be statistically significant ($p < 0.001$). Follow-up Tukey post-hoc tests indicated statistically significant ($p < 0.05$) differences between all pairs of means except for the *jacobs-parncutt* pairs in both corpora. None of the differences in those means are found to be statistically significant. Thus, none of the methods here allow us to state definitively that this accepted enhancement is clearly better than the original. So we can only safely say this about the relative quality of our models per ERR:

1. *ideal*
2. *jacobs* or *parncutt*
3. *random*

This still suggests that ERR is a valid measure for the piano fingering IR task, in all of its tested guises.

Another noteworthy observation is the apparently superior ability of trigram methods to emphasize differences between very good and very bad models. Consider the Hamming MERR of 0.81098 for *ideal-f* and 0.40306 for *random-f*, a difference of 0.40792. This is a surprisingly narrow advantage for the state of the art over one of the worst models imaginable. The "Relaxed" trigram measure provides a wider spread of $0.73802 - 0.13289 = 0.60513$, which is almost 50% higher. This increased spread is statistically significant ($p < 0.001$) and is a desirable attribute in an evaluation method.

5. CONCLUSIONS

The musical fingering problem is best thought of as an information retrieval task, similar to web search. One of its key advantages over static fingering advice like that available in books is its ability to provide a variety of fingering suggestions on demand. The need for such variety of advice is apparent from the ready acceptance of multiple ground truths by the earliest and latest researchers in the domain [1–3].

There are two fundamental ways to frame the fingering problem. The most straightforward is to model the decisions of a single performer. With this framing, to train and to evaluate models, one need only consider passages fingered by this pianist. With few notable exceptions, all prior research in the domain that has striven for quantitative evaluation has implicitly [12–14] or explicitly [15, 16] focused on this (simplified) formulation. Doing so has allowed the field to ignore what is clearly large disagreement among pianists for even the shortest [1] and most routinized [17] of musical segments. Indeed, the seven exercises in the *published* corpus [1] were selected because "at least two distinct, but arguably equally good, fingerings existed for the opening of each piece."

Presenting the "arguably equally good" as needed should be the charter for future computational models of piano fingering. Note that the "ideal" model we describe above is likely far from ideal. As we try to motivate with our enhanced distance functions, some fingerings differ in trivial ways, in ways that explicitly do not render them "distinct" in the sense that Parncutt et al. use the word. Identifying archetypes of distinct *clusters* of fingerings and presenting these archetypes in an optimal order should produce more optimal lists. Clearly, distinct and diverse suggestions make the best lists. Fortunately, an "extension" to the ERR metric [6, §7.4] provides explicit support for this intuitive notion of "diversity." We expect future work to leverage distance measure like those described here for piano to identify diverse clusters and that such extended ERR metrics will better estimate the utility of musical fingering models.

Corpus	Model	Hamming	AdjLong	Trigram	Nuanced	Relaxed
full	*ideal-f*	0.81098	0.85675	0.70206	0.71696	0.73802
full	*jacobs*	0.66966	0.76734	0.51501	0.54404	0.58154
full	*parncutt*	0.66409	0.75903	0.49974	0.52864	0.5402
full	*random-f*	0.40306	0.53721	0.07596	0.10698	0.13289
published	*ideal-p*	0.83794	0.87434	0.77743	0.78681	0.80017
published	*jacobs*	0.6517	0.72936	0.50254	0.51927	0.54028
published	*parncutt*	0.6028	0.70779	0.45561	0.47757	0.50518
published	*random-p*	0.42696	0.58084	0.17156	0.21516	0.25573

Table 4. Mean ERR score using various Δ functions, all phrases treated equally. Mean pairs in shaded rows are not statistically significant.

Again, we have limited the piano fingering problems supported by the methods described above to monophonic phrases played by the right hand. It is straightforward to include the left hand, applying the logic described symetrically in the obvious ways. Either hands may be evaluated using the Python implementation available at `https://github.com/dvdrndlph/pydactyl`. There is also a pragmatic justification for focusing on melodic fingering: it is arguably where pianists are most in need of good advice. Radicioni and Lombardo [18] convincingly demonstrate that chord fingering for guitarists is much less challenging than melodic fingering because the choices are so severely constrained. If anything, chord fingering in piano is even more highly constrained. Having multiple fretboard locations to play a given note, along with more two-dimensional options for placing fingers, affords the guitarist degrees of freedom that are not at the pianist's disposal. Moreover, unlike for guitar, the fingers in a piano chord played with one hand (as is customary or obligatory in the vast majority of cases) must be applied in an ascending order coinciding with the ascending notes. Fingers may be skipped in building the chord, but the order is constrained.

We also note that the metrics can be applied to polyphonic music and will likely perform reasonably well in the face of limited polyphony. However, we must concede that the piano model metrics described here are not comprehensive.

But broadly speaking, ERR is a promising method for evaluating the now clearly framed musical fingering IR task. Its utility must be validated more comprehensively in future work.

6. REFERENCES

[1] R. Parncutt, J. A. Sloboda, E. F. Clarke, M. Raekallio, and P. Desain, "An ergonomic model of keyboard fingering for melodic fragments," *Music Perception*, vol. 14, no. 4, pp. 341–382, 1997.

[2] J. P. Jacobs, "Refinements to the ergonomic model for keyboard fingering of Parncutt, Sloboda, Clarke, Raekallio, and Desain," *Music Perception*, vol. 18, no. 4, pp. 505–511, 2001.

[3] E. Nakamura, Y. Saito, and K. Yoshii, "Statistical learning and estimation of piano fingering," *Information Sciences*, vol. 517, pp. 68–85, 2020.

[4] E. Clarke, R. Parncutt, M. Raekallio, and J. Sloboda, "Talking fingers: an interview study of pianists' views on fingering," *Musicae Scientiae*, vol. 1, no. 1, pp. 87–107, 1997.

[5] D. A. Randolph, J. Badgerow, C. Raphael, and B. Di Eugenio, "Pydactyl: A Python framework for piano fingering," in *Extended Abstracts for the Late-Breaking Demo Session of the 19th International Society for Music Information Retrieval Conference*, Paris, France, 2018.

[6] O. Chapelle, D. Metzler, Y. Zhang, and P. Grinspan, "Expected reciprocal rank for graded relevance," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, Hong Kong, 2009, pp. 621–630.

[7] E. Voorhees and D. M. Tice, "The trec-8 question answering track evaluation," *Proceedings of the 8th Text Retrieval Conference*, 11 2000.

[8] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey, "An experimental comparison of click position-bias models," in *Proceedings of the 1st International Conference on Web Search and Data Mining*, Palo Alto, California, USA, 2008.

[9] A. Kullak, *The Aesthetics of Pianoforte Playing*, 5th ed., H. Bischoff, Ed. New York: G. Schirmer, 1860/1893.

[10] C. Czerny, *160 Kurze Übungen, Op. 821*. Leipzig, Germany: C. F. Peters, 1888. [Online]. Available: http://imslp.org/

[11] D. A. Randolph and B. Di Eugenio, "Easy as abcDE: Piano fingering transcription online," in *Extended Abstracts for the Late-Breaking Demo Session of the 17th International Society for Music Information Retrieval Conference*, New York, 2016.

[12] Y. Yonebayashi, H. Kameoka, and S. Sagayama, "Automatic decision of piano fingering based on a hidden Markov models," in *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, 2007, pp. 2915–2921.

[13] M. Balliauw, "A Variable Neighbourhood Search Algorithm to Generate Piano Fingerings for Polyphonic

Sheet Music," Master of Applied Economic Sciences Thesis, University of Antwerp, 2014.

[14] M. Balliauw, D. Herremans, D. P. Cuervo, and K. Sörensen, "Generating fingerings for polyphonic piano music with a tabu search algorithm," in *Proceedings of the International Conference on Mathematics and Computation in Music*. London: Springer, 2015, pp. 149–160.

[15] R. De Prisco, G. Zaccagnino, and R. Zaccagnino, "A differential evolution algorithm assisted by ANFIS for music fingering," in *Swarm and Evolutionary Computation*, ser. Lecture Notes in Computer Science, L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. A. Zadeh, and J. M. Zurada, Eds., vol. 7269. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 48–56.

[16] E. Nakamura, N. Ono, and S. Sagayama, "Merged-output HMM for piano fingering of both hands," in *Proceedings of the 15th International Society for Music Information Retrieval Conference*, Taipei, Taiwan, 2014, pp. 531–536.

[17] O. Beringer and T. F. Dunhill, *Manual of Scales, Arpeggios, and Broken Chords for Pianoforte*. London: The Associated Board of the Royal Schools of Music, 1989.

[18] D. P. Radicioni and V. Lombardo, "A constraint-based approach for annotating music scores with gestural information," *Constraints*, vol. 12, no. 4, pp. 405–428, apr 2007. [Online]. Available: http://dl.acm.org/citation.cfm?id=1295940.1295956

Compression efficiency and Signal Distortion of common PCA bases for HRTF Modelling

Georgios Marentakis (**Γεώργιος Μαρεντάκης**)[1] and **Josef Hölzl**[2]

[1] *Faculty of Computer Sciences*, **Østfold University College**, Halden, Norway
[2] *Institute of Electronic Music and Acoustics*, **University of Music and Performing Arts Graz**, Graz, Austria

ABSTRACT

Principal Component Analysis (PCA) has been often used for HRTF compression and individualization. However, there is significant variation in how the input matrix on which PCA is applied is constructed. Here, we study the effect of choices on the selection of independent variables, the domain in which impulse responses are represented, the HRTF database used, and possible smoothing on the compression efficiency and the reconstruction quality of the resulting PCA model. Several findings replicate well across different databases. Results point to a benefit for signal compared to space PCA and for using minimum-phase HRIRs or HRTFs. Smoothing HRTFs leads to an increase in compression efficiency and a reduction in spectral distortion and using HRTFs with logarithmic magnitude leads to lower spectral distortion compared to linear.

1. INTRODUCTION

Head Related Transfer Functions (HRTFs) allow designers and engineers to create 3D audio using headphones [1] with applications in virtual and augmented reality. HRTF models that support individualisation, compact representation, and transfer are important as HRTFs are relatively long filters that are specific to individual users and need to be measured for all positions of interest in a relatively resource-intensive process [2].

A compact HRTF model can be reached by decomposing an HRTF set upon a set of orthogonal basis functions and obtaining the related weights (or loadings). Such decompositions can be used to reduce the, typically high, dimensionality of HRTF sets and serve as a basis for compression, individualization, and the investigation of their numerical and perceptual properties. Most often, Principal Component Analysis (PCA) e.g., [3–6] and the Spherical Harmonic Transform e.g., [7–9] have been used for this purpose. More recent approaches focus on deep learning [10].

This article focuses on using PCA for HRTF modelling. It is motivated by the fact that HRTF functions have been arranged in markedly different ways for PCA processing in the literature and aims to investigate the extent to which such differences may affect the compression efficiency and representation ability of the obtained model.

2. BACKGROUND

2.1 HRTFs

The *head-related impulse response (HRIR)* $h(\theta, \phi, t)$ denotes the time domain impulse response for a sound originating at azimuth θ and elevation ϕ measured at or inside the ear-canal. The *head-related transfer function (HRTF)* $H(\theta, \phi, f)$ is the frequency domain representation of the HRIR. HRTFs are recorded using miniature microphones for a subject and source position of interest [2], most commonly, on a dense sampling grid. Frequently, HRTFs are diffuse-field equalized to exclude the ear canal resonance and measurement system response and come in the form of *directional transfer functions* (DTFs).

Binaural cues encoded in the *interaural transfer function*, $ITF = H_L(\theta, \phi, f)/H_R(\theta, \phi, f)$ help localize sounds in the horizontal plane. Monaural cues in the magnitude HRTF spectrum are used for elevation perception and front/back and up/down discrimination [11, 12]. These are spectral peaks and notches between 4 and 16 kHz that are mainly effected by the shape of the outer ear. For example, a prominent 1-octave notch centered between 6 and 11 kHz changes systematically with the vertical source location [13].

Whereas HRTFs incorporate the effects of the whole body, *pinna-related transfer functions (PRTFs)* indicate only the contribution of the pinna and reduce the dependence with respect to azimuth. They can be calculated by applying a 1 ms right window at the beginning of the HRIR signal in order to eliminate reflections by torso and shoulders [14] and then transformed into frequency domain. Such functions are helpful when relating features in the magnitude spectrum to particular anthropometric dimensions. Spectral features below 3 kHz are mainly produced by head diffraction and torso reflections [15].

2.2 Principal Component Analysis

Principal Component Analysis is normally applied onto a two-dimensional matrix, with columns defining the independent variables and rows containing observations. PCA can be calculated directly using the eigendecomposition of the sample covariance matrix $\mathbf{C_Y}$ of the observations or

using the Singular Value Decomposition [16]. The sample covariance matrix $\mathbf{C_Y}$ of a set of observations $\mathbf{Y}$ with M rows of observations and N columns of variables corresponding to a random vector is defined as

$$\mathbf{C_Y} = \mathbf{Y^T}\,\mathbf{Y}\,. \tag{1}$$

$\mathbf{C_Y}$ is as a symmetric, real-valued, square matrix. $\mathbf{Y}$ needs to be centered by subtracting the observation means. The eigenvectors of the covariance matrix $\mathbf{C_Y}$ are also called the principal components of $\mathbf{Y}$. Since $\mathbf{C_Y}$ is symmetric, it is also diagonalizable,

$$\mathbf{C_Y} = \mathbf{V}\,\mathbf{D}\,\mathbf{V}^{-1}\,, \tag{2}$$

with a diagonal matrix $\mathbf{D}$ $(m \times m)$ containing the eigenvalues of $\mathbf{C_Y}$ and $\mathbf{V}$ as an orthonormal eigenvector matrix including the right eigenvectors as columns.

Eigenvectors and eigenvalues may also be obtained through the singular value decomposition (SVD), using which $\mathbf{Y}$ can be written as

$$\mathbf{Y} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^T\,, \tag{3}$$

where $\mathbf{U}$ are $(m{\times}n)$ and $\mathbf{V^T}$ $(n{\times}n)$ orthogonal matrices including *left* and *right* eigenvectors $\mathbf{u}_k$ and $\mathbf{v}_k$, respectively. $\mathbf{S}$ $(n \times n)$ is a diagonal matrix with nonzero non-negative diagonal elements, so that $\mathbf{S} = diag(s_1, \ldots, s_n)$, also known as *singular values*. Note that

$$\mathbf{Y^T}\,\mathbf{Y} = (\mathbf{U}\,\mathbf{S}\,\mathbf{V}^T)^T\,(\mathbf{U}\,\mathbf{S}\,\mathbf{V}^T) = \mathbf{V}\,\mathbf{S}^2\,\mathbf{V}^T\,, \tag{4}$$

Consequently, the square root of the eigenvalues of $\mathbf{YY^T}$ are the singular values (s_k) of $\mathbf{Y}$. The original centered data $\mathbf{Y}$ set can be transformed to the new basis by projecting it on the eigenvector basis $\mathbf{V}$ to obtain the principal component weight (PCW) (or score) matrix $\mathbf{W}$ which can be used for reconstruction.

$$\mathbf{W} = \mathbf{Y}\,\mathbf{V} \text{ and } \mathbf{Y} = \mathbf{W}\,\mathbf{V}^{-1} \tag{5}$$

Assuming that the matrix $\mathbf{Y}$ has a rank r, it follows that $s_k > 0$ for $1 \le k \le r$ and $s_k = 0$ for $(r+1) \le k \le n$ and one can neglect eigenvalues that are very close to zero to reduce the dimensionality. $\mathbf{Y}$ can thus be approximated by reducing the number of eigenvectors involved in the reconstruction.

$$\mathbf{Y}^l = \sum_{k=1}^{l} \mathbf{u}_k\; s_k\; \mathbf{v}_k^T \;+ \bar{\mathbf{Y}}\,, \tag{6}$$

l is commonly chosen by calculating the number of components required to explain, say 90%, of the variance. The variance explained by l components is given by:

$$var(l) = \frac{\sum_{k=1}^{l} s_k}{\sum_{k=1}^{N} s_k} \cdot 100\;[\%]\,, \tag{7}$$

where s_k is the k^{th} singular value, l is the number of a particular PC and N is the total number of components.

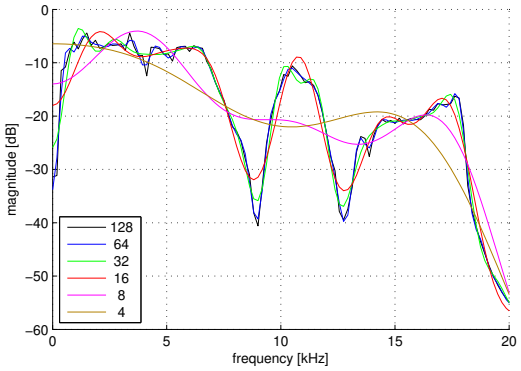


Figure 1. Spectral differences between unprocessed (128 coefficients) and smoothed DTF magnitude spectrum (with 64, 32, 16, 8 and 4 coefficients) in ARI database.

2.3 Modelling HRTFs using PCA

Head-related transfer function sets are mutli-dimensional and typically include the the recorded impulse response for each subject number, direction of sound incidence, and ear. To proceed with PCA, the dataset needs to be placed into a 2D input matrix before calculating principal components and associated weights. Subsequently, the number of required principal components is determined depending on the application [6, 14, 17].

Most of the studies use enough components so as 90% of the variance in the data is explained [3, 15, 18]. Promising results have been obtained when evaluating sound localization with HRTFs that have been reconstructed with a limited set of components subject to the aforementioned variance constraint [3, 19].

Structuring the PCA input matrix: Studies in the literature differ in the HRTF set used, the domain the signal is represented in, and in the way they are transformed into a 2D matrix for PCA.

Some studies apply PCA on HRIRs [5, 15, 20–23]. This is appealing as the time-domain signals maintain delay and phase information and can easily be windowed to isolate the effects of pinna, head, or shoulder. Other studies use minimum-phase HRIRs [5, 15, 23] which do not include direction-dependent delays. Several other studies use HRTF magnitude [3, 4, 17, 24–27] when forming the PCA input matrix and minimum-phase is used when transforming in the time domain. Minimum-phase is used based on the assumption that the original signal phase can be discarded and replaced by a direction-dependent delay.

In the case of HRTFs, PCA has been applied to both linear [4, 28] and logarithmic magnitude spectrum [3, 17, 18, 25, 26, 29] A further difference originates in the number of points that are used in the DTF to estimate the frequency spectrum. More recent studies use the complex spectrum as input to PCA [30, 31].

Most commonly, signal amplitude (time-domain representations) or spectral magnitude (frequency-domain representations) are used as variables in columns and subjects and directions of incidence as observations (rows). Re-

cently, an alternative model has been proposed [6] which uses spatial directions as variables in columns and signal amplitude or spectral magnitude for each subject in rows.

The way the signals from the left and right ears enter the PCA input matrix has also been treated in different ways in literature. Sometimes only one ear is modelled and the second one is considered to be symmetric and therefore duplicated by the modelled one [5, 15]. Alternatively, it can be attempted to use PCA to explain the variability across the two ears. This can be done either by using the time/frequency signals from the second ear as observations in rows [3, 29] in the PCA input matrix.

PCA has not been always performed on the complete set of sound directions in the dataset. Decomposition has been applied on the whole database [3,25], smaller subsets, such as the median [5, 23] or horizontal plane [22, 32], and on single sound directions [15]. The latter approach yields a different set of principal components for each direction, which may not be optimal from a compression perspective. However, as the variability due to direction is not present such an approach allows to focus on individual differences caused by subjects' anthropometry for smaller sets.

A final difference is the HRTF database used to perform the analysis. In general, principal components obtained from different HTRF datasets are consistent as long as the number of measurement directions and subjects is reasonable. This invariance is more evident for components explaining a large amount of variance, as components of smaller variance reflect specificities that might not be shared across datasets. Middlebrooks and Green [26] were among the first who compared basis vectors calculated from their own measurement data (8 subjects, 360 positions) with an existing database by Kistler and Wightman [3] (10 subjects, 265 positions) and indeed confirmed a high correlation between the components, which however decreased with rising principal component order number.

2.4 Summary and Research Questions

The literature review shows that the differences in constructing the PCA input matrix relate to the domain used (time or frequency), the representation (linear or logarithmic magnitude spectrum), the use of minimum-phase HRIRs, the handling of the two ears, and the number of directions analyzed. Given the complexity of modelling HRTFs, it is reasonable to ask what is the impact of choices for the aforementioned parameters on the compression efficiency and the reconstruction potential of the obtained PCA basis. Quite reasonably, researchers would favor an alignment that can represent and re-synthesize the HRTF dataset with the lowest possible number of components and smallest distortion.

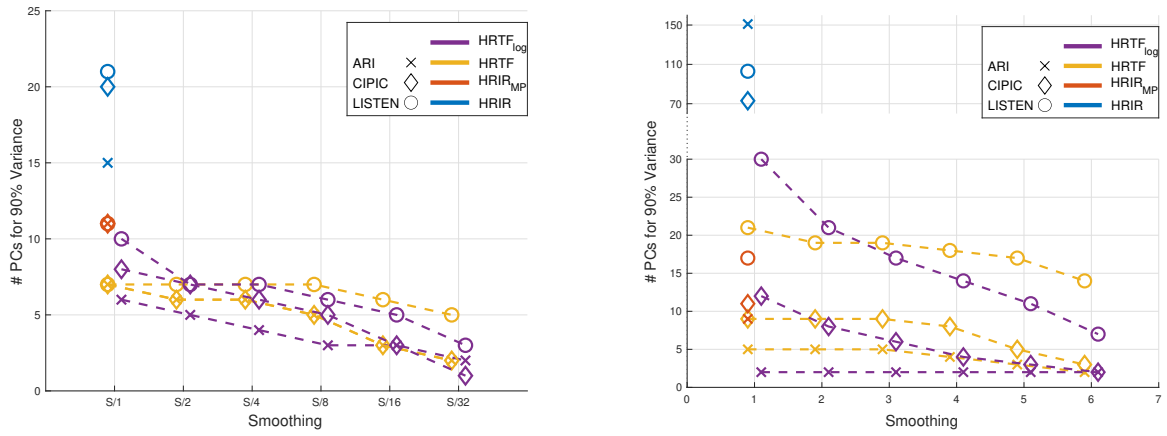
Despite the obvious benefit in identifying an optimal PCA basis, few studies have attempted a direct comparison. Leung and Carlile [19] investigated the PCA compression efficiency and came to the conclusion that the optimal format for PCA decomposition in terms of compression is the linear amplitude form in frequency domain. They used an HRTF dataset of 393 directions. They found that 5 PCs are required for explaining 90% variance when linear magnitude is used; there were less than the number required with logarithmic magnitude. However, the number of subjects or the structure of the PCA input matrix is not clearly described. Takane *et al* [33] extend the work of Liang *et al* [34] and compare four data representations: HRIR, complex spectrum HRTF, linear spectral magnitude HRTF, and log-spectral magnitude HRTF. Sample amplitude (or frequency bin magnitude) appear on input matrix columns and input structures are evaluated based on explained variance, signal distortion, and signal-to-distortion ratio using the KEMAR HATS database [35]. The results confirm an advantage in using representations in the frequency domain but are somewhat inconclusive otherwise. The HRIR database used in this study is this of a dummy head and does not include several or real subjects. Furthermore, the structure of the input matrix structure is not varied to include spatial PCA. Another parameter that has not been considered is the extend to which HRTFs were smoothed. It has been shown that mild spectral smoothing does not affect localization accuracy after reconstruction [36]. Smoothing may have a positive effect on the compression efficiency as perceptually-irrelevant details of HRTF magnitude are smoothed out [37, 38]. For this reason, the smoothing factor is worth including as a parameter in simulations. Overall, a more systematic investigation on the impact of setting up the PCA input matrix on compression efficiency and reconstruction accuracy is attempted here.

3. NUMERICAL EVALUATION

The parameters varied in the evaluation were: HRTF database, the structure of the input matrix, the domain in which the signal was represented, and extent to which HRTFs have been smoothed as explained below. Compression efficiency was evaluated by examining the number of components required to explain 90% of the variance in the input data and by estimating the error in the reconstruction accuracy of the original HRTF set. HRTF reconstruction was evaluated in the frequency domain using the *Spectral Distortion* (SD). For an arbitrary subject s and sound incidence from at θ,ϕ, SD it is calculated by:

$$SD(s,\theta,\phi) = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left[20\log_{10}\frac{|H(s,\theta,\phi,f_j)|}{|\hat{H}(s,\theta,\phi,f_j)|}\right]^2} \quad (8)$$

where $H(s,\theta,\phi,f_j)$ and $\hat{H}(s,\theta,\phi,f_j)$ are measured and estimated HRTF logarithmic magnitudes respectively, and f_j refers to the frequency index, and N is the total number of frequency bins used in the calculation. The synthesized signal is more similar to the measured one when a small SD is obtained. According to [39], the spectral distortion of a reconstructed HRTF should not be greater than 5.7 dB. To measure spectral distortion, the number of PCs used in reconstruction was manipulated from one to all PCs in five steps and the signal distortion was estimated. When HRIRs were used, original and reconstructed HRIRs were



(a) Signal PCA

(b) Spatial PCA

Figure 2. The number of PCs required to explain 90% of variance for signal and spatial PCA in the examined cases. HRIRs were not smoothed and are presented as single points up left on each plot.

transformed in the frequency domain in order to estimate the spectral distortion. The calculation extended over the entire frequency range. Simulations were performed in MATLAB®.

3.1 HRTF Databases

Three open access HRTF databases were used: the Acoustics Research Institute (ARI) HRTF database [40], the LISTEN database from the Institut de Recherche et Coordination Acoustique/Musique [41] and the HRTF database from the University of California at Davis (CIPIC) [42]. ARI contains HRIRs of 256 samples measured at 1550 sound locations and the first 80 subjects were used here. CIPIC includes HRIRs of 200 samples from 45 subjects measured at 1250 sound locations. The LISTEN database HRIRs of 512 samples from 50 subjects and 187 positions. Two subjects were excluded from calculations because impulse responses were not measured for all sound directions. In addition, subject ID 1034 in the LISTEN database resulted in outlying weights and was excluded from the dataset.

3.2 PCA Input Matrix Structure

The first structure (*Signal PCA*) follows a common pattern that has been also used by Kistler and Wightman [3, 29]. Here, signal bins (in frequency or time domain) are the independent variables in columns, while replications for the different subjects and measurement directions are observations in rows. This leads to an input matrix with (subjects × sound directions) rows and (signal bins) columns. The number of rows is doubled if both ears are included and the resulting principal component weights (PCWs) for each subject can be used to recreate the HRIR or HRTF for both ears of each subject and for all directions.

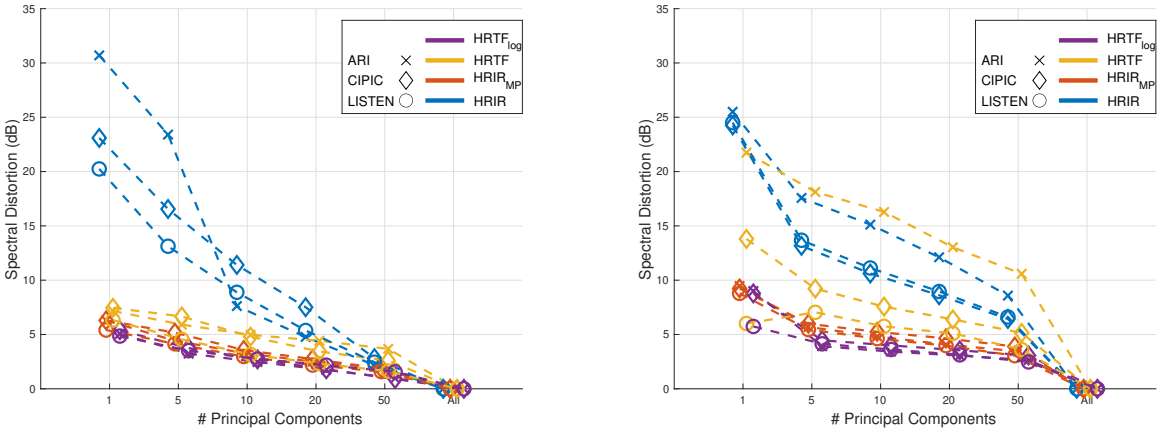
The second structure (*Spatial PCA*) has (subjects × signal bins) rows and (sound directions) columns and was also used by Xie [6, 25]. It lists each sound direction as independent variable in the matrix columns while frequency or time samples from the head-related functions of all subjects are placed as observations in rows. The number of rows is doubled if both ears are included. It has been called spatial PCA because analyzed directions are independent variables placed in columns. The resulting weights can be used to recreate each frequency or time bin for a given position, ear, and subject. In the simulations, HRTFs from both ears were entered as observations in rows for both input structures.

3.3 Signal Domain

For each of the input structures, four signal representations were tested. The first two were in the time domain: the HRIR and the minimum-phase HRIR. The minimum-phase HRIR was used because it allows to remove the direction-dependent initial delay and phase and may reduce the number of components required to represent the signal. Direction-dependent delay can be added after reconstruction in case such a representation is used for compression or individualization. The latter two were in the frequency domain: the magnitude spectrum in either linear or logarithmic amplitude, a common difference in studies applying PCA to HRTFs.

3.4 Smoothing

Smoothing was done by taking the logarithm of the HRTF spectrum, performing FFT, and limiting the number of the Fourier coefficients used to recreate the spectrum, a low-pass filtering operation. Even as few as 16 coefficients within a spectrum of 512 coefficients, a smoothing factor of 1/32 for IR length of 1024 samples, were found to yield satisfactory localization [36]. An example of the output of the smoothing process is shown in Figure 1. Smoothing was only applied when constructing PCA bases using HRTFs and not when using HRIRs.



(a) Signal PCA

(b) Spatial PCA

Figure 3. Spectral Distorion upon reconstruction averaged among subjects, ears, and directions for unsmoothed head related functions in the conditions examined in the simulations.

4. RESULTS

The results of the simulation ar epresented next starting from the number of components required to explain 90% of the input matrix variance and following up with the spectral distortion results.

4.1 Compression Efficiency

By observing Figure 2, it can be seen that the input matrix structure has a considerable impact on compression efficiency. For the CIPIC and LISTEN databases, most efficient is signal PCA followed by spatial PCA. This is consistent across smoothing factors. For the ARI database, spatial PCA using either linear or log HRTF magnitude yields a compression efficiency that is higher than signal PCA.

Quite clearly, the HRIR representation requires most components, irrespective of whether signal or spatial PCA is performed. Taking the minimum-phase HRIR results in a significant reduction in the number of PCs required to explain 90% of the variance which makes using minimum-phase HRIRs comparable to PCA using unsmoothed HRTFs in terms of compression efficiency. This result is consistent across the databases examined here.

The number of PCs required by frequency domain representations is reduced significantly due to the application of spectral smoothing. Each time the Fourier coefficients used in spectral reconstruction are halved, a significant reduction in the number of components required to explain 90% variance is observed.

The impact of a linear or logarithmic magnitude representation in the frequency domain is not as clear-cut. For unsmoothed HRTFs, a small advantage for linear amplitude representation is registered for the LISTEN and CIPIC database for both signal and space PCA. As long as smoothing is applied, the situation is reversed and the logarithmic representation results in a smaller number of required components. For the ARI database, a small advantage for logarithmic magnitude representation appears which remains consistent as smoothing is applied.

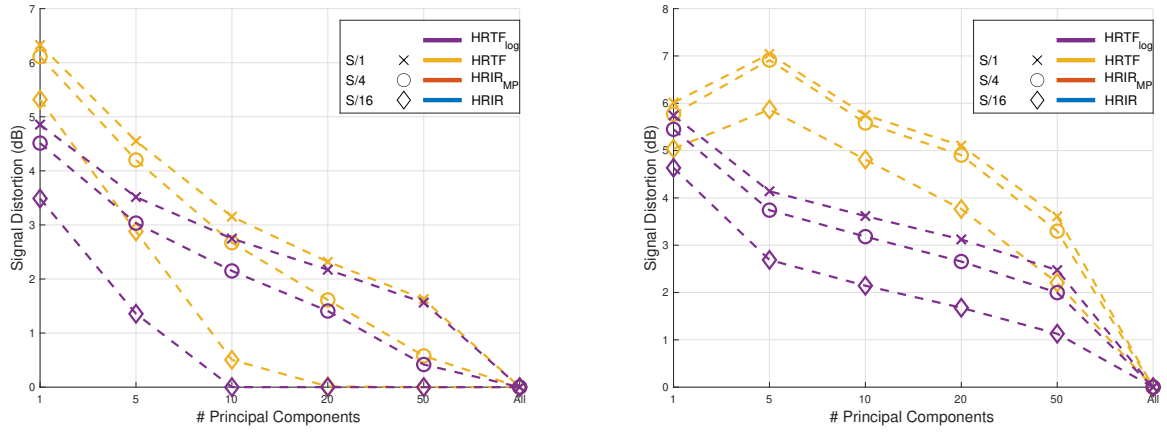
The number of components required to explain 90% of the variance is consistent across databases for the signal PCA. However, it varies considerably when spatial PCA is considered and the number of required components is doubled for ARI to CIPIC and then the LISTEN database.

4.2 Spectral Distortion

By observing Figure 3, it can be seen that spectral distortion results are in agreement with the compression efficiency observations. Spectral distortion was highest when HRIRs were used in the input matrix. Spectral distortion was reduced significantly when minimum-phase HRIRs or HRTFs were used. This result is consistent across databases.

For signal PCA, signal distortion is lowest when minimum-phase HRIRs and the logarithmic HRTFs are used in the PCA input matrix and falls below 5 dB as soon as 5 components are used for reconstruction. Signal PCA with with linear amplitude HRTFs result in an overall higher spectral distortion. This result is consistent across databases.

For spatial PCA, again PCA with minimum-phase HRIRs or logarithmic magnitude HRTFs result in the lowest spectral distortion which again falls below 5 dB as long as at least 5 components are used for reconstruction. Spectral distortion is highest for the ARI database for the linear magnitude HRTFs and the HRIRs compared to the rest but the differences among databases were smaller for minimum-phase HRIRs and log-magnitude HRTFs. Interestingly, achieving a spectral distortion below 5 dB requires a higher number of components than the one required to explain 90% of the variance in the ARI database for the spatial PCA. By observing Figure 4, it can be seen that smoothing does seem to reduce spectral distortion. This effect was consistent across databases input structures and appeared both for linear and log-magnitude HRTFs.



(a) Signal PCA

(b) Spatial PCA

Figure 4. Spectral Distorion upon reconstruction averaged across subjects, ears, and sound directions for smoothed head related functions taken from the LISTEN database.

5. DISCUSSION

In this study, we investigated the impact of choices in the design of the input matrix used to analyze head related impulse responses or head related transfer functions using principal component analysis. The impact of matrix structure (signal or spatial PCA) and the signal domain (time or frequency) was manipulated. For the time-domain HRIRs both raw and minimum-phase HRIRs were compared, while for the frequency domain linear and logarithmic amplitude was compared. Finally, for frequency domain representations, the impact of spectral smoothing was also considered. Three different HRTF databases were used in the analysis. The number of components required to explain 90% of the variance and the spectral distortion upon reconstruction were used as objective measures for the purpose of comparison.

The difference in the number of components required to explain 90% of the variance among the databases used here was small for signal PCA compared to spatial PCA. The variable to observation ratio for spatial PCA was 0.27 for ARI, 0.15 for CIPIC, and 0.015 for LISTEN, while for signal PCA it was 0.013 for LISTEN 0.0009 for CIPIC, and 0.0005 for ARI. The signal PCA configurations have a better variable to observation ratio which may explain the better consistency of the results across databases for signal compared to spatial PCA.

For the LISTEN and CIPIC databases, fewer components were required to represent 90% of the input matrix variance and the resulting spectral distortion upon reconstruction was lower for signal PCA in comparison to spatial PCA. However, a lower number of components was required to account for 90% of the variance when analyzing the ARI database using spatial compared to signal PCA required, which was even lower when the log-magnitude spectrum was used. The number of components required for spatial PCA in the ARI database was consistently small even when the number of database locations used was reduced and the variable to observation ratio improved. However, the suggested number of components yielded increased spatial distortion upon reconstruction and would need to be increased to keep spectral distortion below 5 dB. Further incestigation is required to confirm if Spatial PCA can lead to an effective PCA basis and to explain the discrepancy among databases.

Overall, the raw HRIR representation was the most inefficient both in terms of compression efficiency and in terms of the resulting Signal Distortion upon reconstruction, in agreement with observations in the literature [33]. Removing the direction dependent delay and phase from the signals by taking the minimum-phase impulse response reduced the number of components and the spectral distortion upon reconstruction dramatically and made principal component analysis as efficient as with input matrices using spectral HRTFs.

A beneficial effect of smoothing for PCA using HRTFs was observed which improved compression efficiency and reduced spectral distortion. As smoothed HRTFs have been found to provide good sound localization, this may be a good option to consider in future applications of principal component analysis up to the point where localization is not affected and coloration does not appear [36]. It would be interesting to examine if a similar result would have been observed if the HRIRs were smoothed using a low pass filter in the time-domain but this was not investigated here.

Concerning the impact of a linear or logarithmic representation for PCA analysis based on spectral HRTF data the results are not as clear-cut. On the one hand, there is a tendency for lower number of components for representing 90% of the variance for the linear amplitude, as also mentioned by [19, 33, 38] but this advantage tends to be cancelled as long as smoothing is applied. Furthermore, the logarithmic representation leads to a lower spectral distortion. It appears therefore that the logarithmic representation may be more efficient if PCA is to be performed on HRTFs and both criteria are considered.

6. CONCLUSION

We presented a study that investigated the impact of HRTF database, input structure (signal or space), signal domain (time or frequency), and HRTF smoothing on the compression efficiency and the spectral distortion upon reconstruction when modelling HRTFs using Principal Component Analysis. The results of the numerical simulations show that signal PCA has a better compression efficiency (2/3 databases) and lower spectral distortion (3/3 databases) upon reconstruction. Furthermore, using HRIRs as input to PCA leads to worse compression efficiency and higher spectral distortion compared to HRTFs. Using minimum-phase HRIRs compensates for this discrepancy. Minimum-phase HRIRs lead to comparable compression efficiency and spectral distortion compared to HRTFs. Applying smoothing to HRTFs leads to an increase in compression efficiency and a reduction in spectral distortion for all databases used. Logarithmic magnitude leads to lowest spectral distortion when using HRTFs while compression efficiency depends on the database used.

7. REFERENCES

[1] D. R. Begault, *3-D sound for virtual reality and multimedia*. Morgan Kaufmann Pub, 1994.

[2] S. Li and J. Peissig, "Measurement of head-related transfer functions: A review," *Applied Sciences*, vol. 10, no. 14, p. 5014, 2020.

[3] D. J. Kistler and F. L. Wightman, "A model of head-related transfer functions based on principal components analysis and minimum-phase reconstruction," *The Journal of the Acoustical Society of America*, vol. 91, no. 3, pp. 1637–1647, 1992.

[4] J. Qian and D. A. Eddins, "The role of spectral modulation cues in virtual sound localization," *The Journal of the Acoustical Society of America*, vol. 123, no. 1, pp. 302–314, 2008.

[5] S. Hwang, Y. Park, and Y.-s. Park, "Modeling and customization of head-related impulse responses based on general basis functions in time domain," *Acta Acustica united with Acustica*, vol. 94, no. 6, pp. 965–980, 2008.

[6] M. Zhang, Z. Ge, T. Liu, X. Wu, and T. Qu, "Modeling of individual hrtfs based on spatial principal component analysis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 785–797, 2020.

[7] M. J. Evans, J. A. Angus, and A. I. Tew, "Analyzing head-related transfer function measurements using surface spherical harmonics," *The Journal of the Acoustical Society of America*, vol. 104, no. 4, pp. 2400–2411, 1998.

[8] W. Zhang, T. D. Abhayapala, R. A. Kennedy, and R. Duraiswami, "Modal expansion of hrtfs: Continuous representation in frequency-range-angle," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009, pp. 285–288.

[9] ——, "Insights into head-related transfer function: Spatial dimensionality and continuous representation," *The Journal of the Acoustical Society of America*, vol. 127, no. 4, pp. 2347–2357, 2010.

[10] R. Miccini and S. Spagnol, "Hrtf individualization using deep learning," in *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE, 2020, pp. 390–395.

[11] J. Blauert, "Spatial hearing: The psychophysics of human sound localization," 1983.

[12] E. H. A. Langendijk and A. W. Bronkhorst, "Contribution of spectral cues to human sound localization," *The Journal of the Acoustical Society of America*, vol. 112, no. 4, pp. 1583–1596, 2002.

[13] B. Zonooz, E. Arani, K. P. Körding, P. R. Aalbers, T. Celikel, and A. J. Van Opstal, "Spectral weighting underlies perceived sound elevation," *Scientific reports*, vol. 9, no. 1, p. 1642, 2019.

[14] S. Spagnol, M. Geronazzo, and F. Avanzini, "Fitting pinna-related transfer functions to anthropometry for binaural sound rendering," in *2010 IEEE International Workshop on Multimedia Signal Processing*. IEEE, 2010, pp. 194–199.

[15] K. H. Shin and Y. Park, "Enhanced vertical perception through head-related impulse response customization based on pinna response tuning in the median plane," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 91, no. 1, pp. 345–356, 2008.

[16] I. T. Jolliffe, *Principal component analysis*. Springer Verlag, 2002.

[17] S. Xu, Z. Li, and G. Salvendy, "Identification of anthropometric measurements for individualization of head-related transfer functions," *Acta Acustica united with Acustica*, vol. 95, no. 1, pp. 168–177, 2009.

[18] W. L. Martens, "Principal components analysis and resynthesis of spectral cues to perceived direction," in *Proc. Int. Computer Music Conf., Champaine-Urbana, IL*, 1987.

[19] J. Leung and S. Carlile, "Pca compression of hrtfs and localization performance," in *International Workshop on the Principies and Applications of Spatial Hearing*, 2009.

[20] Z. Wu, F. H. Chan, F. Lam, and J. C. Chan, "A time domain binaural model based on spatial feature extraction for the head-related transfer function," *The Journal of the Acoustical Society of America*, vol. 102, no. 4, pp. 2211–2218, 1997.

[21] D. W. Grantham, J. A. Willhite, K. D. Frampton, and D. H. Ashmead, "Reduced order modeling of head related impulse responses for virtual acoustic displays," *The Journal of the Acoustical Society of America*, vol. 117, no. 5, pp. 3116–3125, 2005.

[22] K. J. Fink and L. Ray, "Tuning principal component weights to individualize hrtfs," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 389–392.

[23] S. Hwang, Y. Park, and Y.-s. Park, "Customization of Spatially Continuous Head-Related Impulse Responses in the Median Plane," *Acta Acustica united with Acustica*, vol. 96, no. 2, pp. 351–363, Mar. 2010.

[24] G. Grindlay and M. A. O. Vasilescu, "A multilinear (tensor) framework for hrtf analysis and synthesis," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, vol. 1. IEEE, 2007, pp. I–161.

[25] B.-S. Xie, "Recovery of individual head-related transfer functions from a small set of measurements," *The Journal of the Acoustical Society of America*, vol. 132, no. 1, pp. 282–294, 2012.

[26] J. C. Middlebrooks and D. M. Green, "Observations on a principal components analysis of head-related transfer functions," *The Journal of the Acoustical Society of America*, vol. 92, no. 1, pp. 597–599, 1992.

[27] J. Chen, B. D. Van Veen, and K. E. Hecox, "A spatial feature extraction and regularization model for the head-related transfer function," *The Journal of the Acoustical Society of America*, vol. 97, no. 1, pp. 439–452, 1995.

[28] M. Rothbucher, M. Durkovic, H. Shen, and K. Diepold, "Hrtf customization using multiway array analysis," in *2010 18th European Signal Processing Conference*. IEEE, 2010, pp. 229–233.

[29] F. Wightman and D. Kistler, "Localization of virtual sound sources synthesized from model hrtfs," in *Final Program and Paper Summaries 1991 IEEE ASSP Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, 1991, pp. 0_51–0_52.

[30] O. A. Ramos and F. C. Tommasini, "Magnitude modelling of hrtf using principal component analysis applied to complex values," *Archives of Acoustics*, vol. 39, 2014.

[31] R. Bomhardt, H. Braren, and J. Fels, "Individualization of head-related transfer functions using principal component analysis and anthropometric dimensions," in *Proceedings of Meetings on Acoustics 172ASA*, vol. 29, no. 1. Acoustical Society of America, 2016, p. 050007.

[32] K. J. Fink and L. Ray, "Individualization of head related transfer functions using principal component analysis," *Applied Acoustics*, vol. 87, pp. 162–173, 2015.

[33] S. Takane, "Effect of domain selection for compact representation of spatial variation of head-related transfer function in all directions based on spatial principal components analysis," *Applied Acoustics*, vol. 101, pp. 64–77, 2016.

[34] Z. Liang, B. Xie, and X. Zhong, "Comparison of principal components analysis on linear and logarithmic magnitude of head-related transfer functions," in *2009 2nd International Congress on Image and Signal Processing*. IEEE, 2009, pp. 1–5.

[35] W. G. Gardner and K. D. Martin, "Hrtf measurements of a kemar," *The Journal of the Acoustical Society of America*, vol. 97, no. 6, pp. 3907–3908, 1995.

[36] A. Kulkarni and H. S. Colburn, "Role of spectral detail in sound-source localization," *Nature*, vol. 396, no. 6713, pp. 747–749, 1998.

[37] J. Breebaart, F. Nater, and A. Kohlrausch, "Spectral and spatial parameter resolution requirements for parametric, filter-bank-based hrtf processing," *Journal of the Audio Engineering Society*, vol. 58, no. 3, pp. 126–140, 2010.

[38] J. Breebaart, "Effect of perceptually irrelevant variance in head-related transfer functions on principal component analysis," *The Journal of the Acoustical Society of America*, vol. 133, no. 1, pp. EL1–EL6, 2013.

[39] T. Nishino, S. Kajita, K. Takeda, and F. Itakura, "Interpolating head related transfer functions in the median plane," in *Proceedings of the 1999 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. WASPAA'99 (Cat. No. 99TH8452)*. IEEE, 1999, pp. 167–170.

[40] P. Majdak, Y. Iwaya, T. Carpentier, R. Nicol, M. Parmentier, A. Roginska, Y. Suzuki, K. Watanabe, H. Wierstorf, H. Ziegelwanger *et al.*, "Spatially oriented format for acoustics: A data exchange format representing head-related transfer functions," in *Audio Engineering Society Convention 134*. Audio Engineering Society, 2013.

[41] O. Warusfel. (2003) Listen hrtf database. [Online]. Available: http://recherche.ircam.fr/equipes/salles/listen

[42] V. R. Algazi, R. O. Duda, D. M. Thompson, and C. Avendano, "The cipic hrtf database," in *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No. 01TH8575)*. IEEE, 2001, pp. 99–102.

LIVE RAY TRACING AND AURALIZATION OF 3D AUDIO SCENES WITH VARAYS

Émile OUELLET-DELORME (eodelorme@sat.qc.ca) [1], **Harish VENKATESAN**[1],
Emmanuel DURAND (edurand@sat.qc.ca) [1], and **Nicolas BOUILLOT** (nbouillot@sat.qc.ca) [1]

[1] *Metalab*, **Society for arts and technology**, Montreal, Quebec Canada

ABSTRACT

We present vaRays, our open-source room acoustic modelling library (C++/Python) that uses acoustic ray tracing to model acoustic interactions of sources, listeners and the virtual environment and auralize the scene in 3D using an ambisonic convolver. The paper describes the different sub-systems of the library: the ray tracer, the ambisonic impulse response encoder and the auralizer, as well as the design choices targeting live auralization of 3D scenes. We demonstrate the use of vaRays with navigation in a 3D model in which sound sources and listener position are communicated to vaRays from a 3D game engine through network message (OSC). We also demonstrate the use of vaRays as a live acoustical simulation system in our large dome, simulating continuously varying acoustic applied to acoustical musical instrument captured with a microphone.

1. INTRODUCTION

At the forefront of live spatial audio research and applications, the *object-based spatialization* [1] paradigm, where sources are located in space using positioning data (azimuth, distance and elevation with reference to the listener), is leading authoring tools and spatial audio compositions. In this paradigm, the spatialization of sound is recreated synthetically by considering the sound sources as points located in space, without the knowledge of the geometry of the simulated space. Inclusion of geometry specific acoustic calculations such as occlusion, sound transfer through materials and reverberation are left to the user.

Fortunately a geometrical scene description (a 3D model), annotated with acoustical properties for each surface, enables the design of algorithm and tools for live computation of spatial audio. More precisely, real-time modelling of the acoustics of a scene at regular intervals is investigated here for virtual reality and immersion applications. Several geometrical acoustic techniques have been described in the past [2] to find acoustic paths between a source-listener pair, such as image-source method, ray tracing and beam tracing. With these paths, a directional specification of the delays and the power of the reverberations constitute what is called the Impulse Response (IR) of the location for the listener-sound source pair. More precisely, we are interested in real-time auralization, i.e. the process of making audible, by physical or mathematical modelling, the sound field of a sound source in a geometrical space for a specific listener position [3]. This auralization strategy has already been experimented and presented in the literature [2, 4]. It is usually based on live computation of Room Impulse Response (RIR), used to live feed a convolution reverberation.

In this paper, we introduce vaRays, our open-source room acoustic simulation C++ library that aims to address some issues pertaining to acoustic simulation. vaRays uses real-time ray tracing to model the transmission of acoustic energy and its interactions with the environment in a virtual scene. Some of these interactions include absorption by air molecules (air attenuation) and by the surfaces in the virtual environment (material absorption), specular and diffuse reflections, diffraction and transmission through surfaces. Ray tracing is a geometrical acoustic modelling technique [2] where these interactions are modelled by launching rays or packets of energy that travel in straight lines, much like in ray optics to identify the paths through which acoustic energy is transmitted from the sources in a scene to the listener. In vaRays, these modelling results are used to create spatial RIRs in ambisonic format which are then used in the auralization of the virtual scene.

vaRays was first mentioned in [5] as a supplementary tool during its early stages. Here, we discuss vaRays in detail describing the different sub-systems, illustrating and elaborating our design choices. The next section gives an overview of the vaRays pipeline, starting with the scene geometry all the way to auditory output. We provide an overview of vaRays and discuss in detail the different subsystems of vaRays in Sec. 2, describe in brief a typical usage along with tools built at the Society for arts and technology [SAT] in Sec. 3 and finally conclude and discuss some future pathways in Sec. 4.

2. THE SUB-SYSTEMS OF VARAYS

The vaRays library is primarily written in modern C++ in order to provide acoustic simulation and auralization in real-time on Unix based operating systems. The vaRays software package also includes a Python wrapper for all the important classes for ease of application development and compatibility with popular game engines and 3D animation tools such as Godot and Blender.

vaRays is mainly organized into three mutually exclusive

sub-systems based on the operations performed, namely:

1. **vaRays Context**, which is responsible for tasks related to the ray tracing operation such as parsing the geometry of the environment, launching and tracing rays and providing simulation results in the form of sound events for each source-listener pair in the scene,

2. **vaRays Encoder**, which is responsible for generating ambisonic RIRs from the ray tracing results,

3. **vaRays Auralizer**, which performs live convolution of audio streams with the RIRs generated by the vaRays encoder.

In the following sections, we describe each of these subsystems in detail.

2.1 vaRays Context

Acoustic ray tracing is in many ways similar to the ray tracing techniques employed in computer graphics to compute illumination of objects by light sources, with the main difference that sound takes a finite measurable time to travel from a source to a listener, while light can be considered to travel instantaneously for all practical purposes. We use ray tracing to find acoustic paths between the sources and listeners in real time, mainly using methods described in [6]. Since ray tracing is a stochastic technique, the greater the amount of rays thrown, the more accurate the RIRs will be, but the slower the calculations will be as well. New RIRs are generated periodically for every source and listener pair that exists in the scene. The following information is required in order to perform ray tracing:

1. **Sources and listeners:** At least one source and one listener are necessary for the ray tracing to occur. For each pair of source and listener that exists within the scene, one set IRs will be generated (for achieving spatialization - explained in Sec. 2.2). The listeners correspond to coordinates in a scene and do not have a volume. The sources are also coordinates, but vaRays considers them to be spheres with a radius that can be modified by the user.

   The main reason for the sources having a volume is to avoid missing rays that are close to the source because of the stochastic nature of ray-tracing. This was first introduced by Vorlander [7]. The larger the volume, the more energy will be picked up by the source. Since volume is only used to calculate energy, it does not interact with the scene. As such, the volume of the source mostly has an incidence on the decay of the energy relative to the distance. A larger volume can, for example, help mitigate unexpected spikes of volume when the listener ventures too close to the source.

   Each source also has its own energy level that will be split between each thrown ray. That energy can have a different level depending on frequency, though the default values are evenly distributed. All sources are considered to be omnidirectional. Directivity for sources is not yet supported, but will be in a future update.

2. **Room geometry:** The geometry is the environment in which rays will be thrown. It is given to the ray tracing engine and updated when required. The geometry contains all surfaces, whose interaction with the incident acoustic rays will be simulated. It however does not contain the sources and listeners as we do not consider collisions with these objects possible during the ray tracing. Geometries are currently loaded using Wavefront OBJ files and do not consider vertex normals (collisions will occur on both faces of a surface regardless).

3. **Materials:** All surfaces in the environment are made of materials which are described by absorption and scattering coefficients in different frequency bands. The scattering coefficients correspond to the proportion of energy that will be diffused during reflections as opposed to reflected specularly. Currently, this coefficient will always be the same regardless of the incident angle of a ray. The absorption coefficient corresponds to the amount of energy that will be lost when a reflection occurs on that surface.

A large amount of stochastic rays are thrown periodically from every listener in a scene. Each of those rays carries an energy map for different frequencies. When a ray hits a surface or a diffraction area, more stochastic rays will be thrown from that point and the energy carried by the initial ray that remains after absorption will be split between children rays. This happens recursively until a maximum reflection order or a minimum energy threshold has been reached for an individual ray. The sources and listeners are not objects in the scene. In order to know if sound reaches its destination, visibility tests are computed from every collision points. Here are the different steps to the ray tracing:

2.1.1 Ray trajectory

It is recommended to throw the rays from listeners, rather than from the sources, which might be unintuitive. The main reason is that when using the diffuse rain method (see Fig. 3) with more than one source in the scene, tracing the rays backwards will greatly reduce the amount of rays to be traced. This will be described with more details in Sec. 2.1.3. While it saves lots of rays from being thrown, this method does have an important drawback. Specular reflections are reciprocal, which means that they will be the same regardless of the point from which we throw the rays. Diffuse reflections are not. While this error is not significant for most room shapes, the possibility of throwing the rays forward always remains if tests show that it is.

2.1.2 Initial rays and direct sound

For each source, the amount of energy emitted from the listener is fixed. Before launching rays, the proportion of energy carried by the direct sound reaching a source with line-of-sight from the listener is calculated as a function of

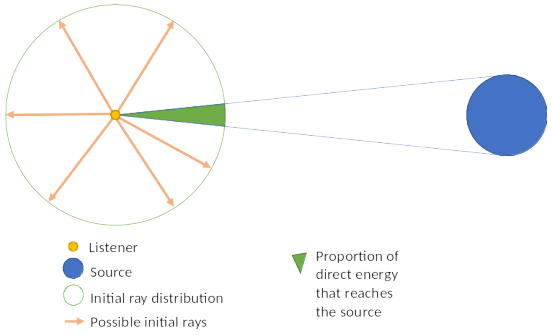


Figure 1. Direct sound computation for a pair of listener and source.

the solid-angle that it subtends (see Fig. 1). The remaining energy is then distributed equally among the rays that are launched. It is important to note here that no two sources in the scene share the energy launched from the listener, i.e., from the perspective of a given source, all the energy launched from the listener either reach the source or is lost due to absorption, but not received by any other source in the scene.

After the direct sound is computed, the initial rays are thrown. As we throw the ray backwards, rather than containing energy, each ray represents a proportion of receivable energy from the listener. Those rays are sent from a specific point which corresponds to the coordinates of a listener in the scene. The rays are thrown in random directions (uniformly distributed on the surface of a sphere), and the receivable energy is evenly distributed between them. When a ray is thrown, the ray tracer will return the coordinates of the first collision point it reaches (if any) and the material that was defined for that surface.

2.1.3 Reflections

When a thrown ray hits a surface, a reflection event occurs (see Fig. 2). First, the proportion of receivable energy is attenuated by the absorption coefficient of the surface. The remainder is then split in two parts using the scattering coefficient of the surface, the specular and diffuse energies.

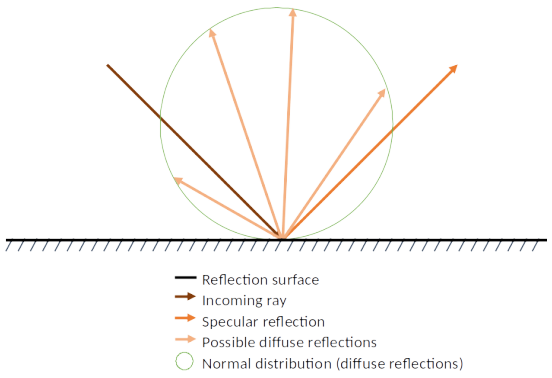


Figure 2. Reflection model used in vaRays.

1. **Specular:** A specular reflection will always occur from every reflection point, and it carries the full proportion of receivable specular energy that was calculated previously.

2. **Diffuse:** The amount of diffuse rays sent from every reflection point depends on the user properties (see Fig. 2), and it can change depending on the reflection order. The diffuse proportion of receivable energy at the collision point is evenly split between each diffuse ray that will be created. The proportion of energy that is diffusely reflected from the collision point is evenly split between the Lambertian diffuse rays whose directions are normally distributed about the normal to the surface. This distribution could be changed or modified in the future.

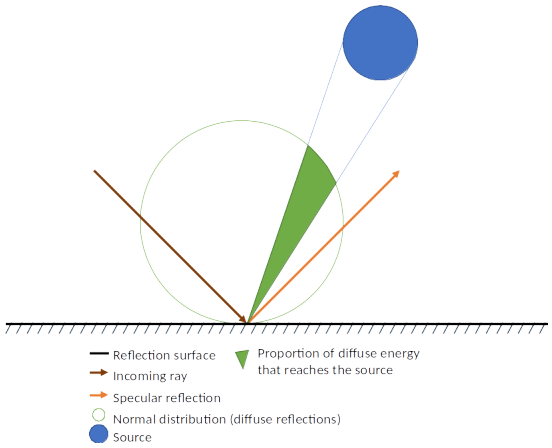


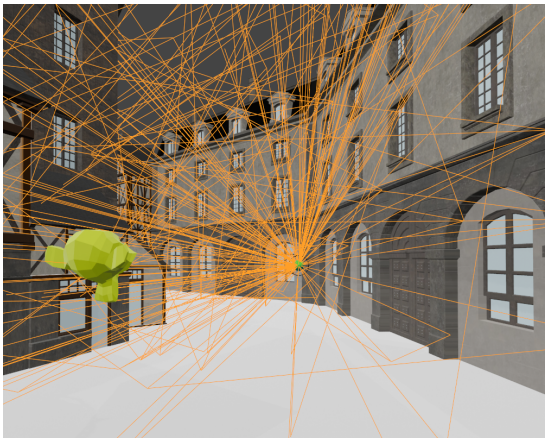
Figure 3. Diffuse rain method.

Before any reflected rays are thrown, a visibility test with each source in the scene is done, similar to what was done to calculate the direct energy. This method is called diffuse rain [6]. If a source is visible, the proportion of specular and diffuse energy that reaches the source is calculated. Since the specular energy only has one specific direction, the proportion that reaches the source is either 0 or 1. In order to calculate the proportion of diffuse energy that reaches the source, we use a variation of the same method we did to calculate the direct energy that reaches the source. The energy that reaches the source is saved (and will contribute to the IR), then subtracted from the sum of energy that will get reflected from that point. Fig. 4 shows the acoustic reflection paths computed by tracing stochastic rays in a virtual environment.

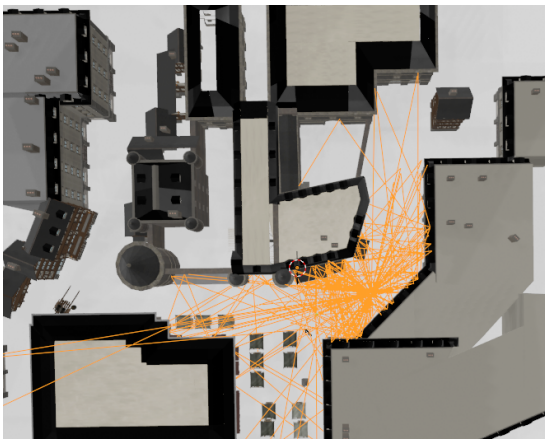
2.2 vaRays Encoder

In this section and Sec. 2.3, we explain our approach to auralization, the process of translating simulation results to auditory output. There are majorly two known methods for auralizing simulation results:

1: delay-line based approach, where a single delay-line (or a single output of a multitap delay-line) rep-



(a) In-world viewcaption



(b) Top view

Figure 4. Acoustic paths (orange lines) identified between a source (green ball) and a listener (yellow monkey head) in a model of an 18th century Parisian city block, visualized on Blender.

resents a single acoustic path along with the filter blocks for simulating material absorption,

2: convolution based approach, where all acoustic paths are reduced to a finite set of IRs that are convolved with dry/unprocessed audio signal from the source.

The first approach is very effective in auralizing dynamic scenes (moving listeners and sources) by updating delay lengths and filter coefficients at audio-rate [4], thus being able to auralize acoustic phenomena such as Doppler shift. However, it suffers from the major disadvantage of being computationally expensive for dense simulations with large number of acoustic paths.

The second approach is very efficient in auralizing dense simulations (computational cost is fixed for fixed IR lengths) but is not effective in auralizing dynamic scenes and requires filter update mechanisms. In vaRays, we take a hybrid approach exploiting both techniques. The computation of IRs for convolution is discussed in detail in this section and the auralization system is explained in Sec. 2.3.1.

Assuming all interactions between the acoustic rays and the acoustic surfaces in the scene are linear and time-invariant (LTI), static source-room-listener systems can be considered as LTI systems described by RIRs. For any given scene configuration, the vaRays context provides us with the following information about acoustic paths between each source-listener pair, using which the vaRays encoder computes spatialized RIRs in higher-order-ambisonic (HOA) format:

1. **Path distance:** The distance travelled by the ray from the listener to the sources after reflections.

2. **Direction of departure:** The initial direction in which the ray was launched from the listener. This information is essential for spatialization of the RIR.

3. **Energy profile:** The energy distribution of the ray over pre-determined frequency bands (explained in greater detail later in this section).

An RIR is essentially a series of delayed, filtered and scaled Dirac impulses corresponding to the acoustic paths between a source and a listener (currently in vaRays, the phase evolution of the impulses during propagation is not considered). The vaRays encoder essentially builds convolution kernels (signals with which the dry audio signal from the source is convolved) from the time-energy-space echograms produced by the ray tracer. We employ filter updating techniques to auralize dynamic scenes from static RIRs (explained in Sec. 2.3). In real-time auralization of dynamic scenes, computing these spatial RIRs with minimal time delay is critical in maximizing immersion [8]. We mainly considered two approaches to generating RIRs, but we only discuss the approach that better suits real-time auralization requirements in this paper. Detailed account of the experiments and choice of approach can be found here [1] .

Each ray reaching a source from the listener is filtered by the surfaces participating in its propagation due to their acoustic properties (frequency dependent absorption, transmission and scattering) and attenuated due to air absorption. The vaRays material property database (mostly derived from [8] and [4]) contains properties defined for 8 octave bands between the range of 125 - 16000 Hz. Air absorption coefficients are computed [9] for these frequency bands on start-up and are saved for use during auralization. In vaRays, we take a source-filter approach to generate RIRs by supplying excitation pulses as input to a perfect-reconstruction filter-bank.

We make use of a digital Linkwitz-Riley [10] crossover filter tree that consists of filter pairs as shown in Fig. 5. The Linkwitz-Riley filter pair is made of a 4th-order Butterworth highpass and lowpass filter with their critical frequencies at the crossover point (the frequency separating two adjacent bands). This filter structure is known for its orthogonality property and is widely used in loudspeaker

[1] `https://gitlab.com/sat-metalab/vaRays/-/tree/master/Playground/IR_Generation`, accessed May 2021.

crossover circuits and digital multi-band processing. In our context, we use the LR filter tree as the filter in the source-filter model with 8 inputs corresponding to the 8 octave bands.

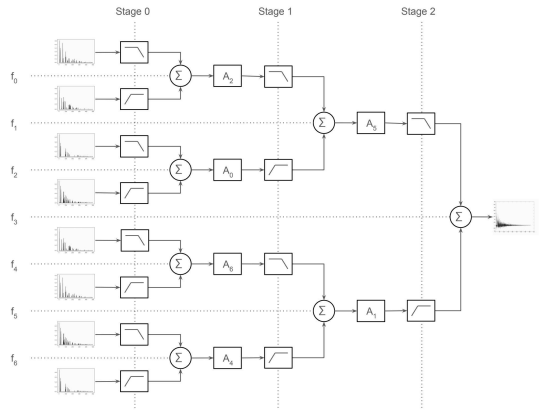


Figure 5. Block diagram of the Linkwitz-Riley crossover filter tree used for generating RIRs in vaRays. Blocks with flat line followed by decreasing ramp are lowpass filters and those with increasing ramp followed by flat line are highpass filters. The inputs to the filter tree are excitation impulse trains (left extreme) corresponding to bands between the crossover frequencies indicated by dotted lines, and the output is the RIR (right extreme).

An LR filter tree with N stages (here $N = 3$) has 2^N inputs for a different frequency band (8 inputs). Each stage I of the filter tree is responsible for band-merging at crossover frequencies f_i where $i = I + n * \left(2^{I+1}\right)$ and $n \in 0, 1, .., 2^{N-(I+1)} - 1$. The all-pass filters (denoted by the letter A in Fig. 5) compensate for the phase shifts caused by the LR filter pairs on the other branches of the tree.

For a mono IR, the vaRays encoder first generates a train of Dirac impulses delayed by an amount proportional to the propagation path of the rays and scaled proportional to the energy contained in each frequency band, giving a total of 8 excitation impulse trains to be fed synchronously to the corresponding inputs of the LR filter tree. The output of the filter-tree is a mono IR. For an ambisonic RIR of order N_{ambi}, the same structure is repeated for each of the $(N_{ambi} + 1)^2$ ambisonic channels with the impulses being scaled additionally by the ambisonic coefficients for the direction of departure of the ray from the listener. vaRays currently supports only up to 6th order SN3D ambisonic format.

This approach to generating RIRs scales very well with increasing density of simulation with increasing number of acoustic paths for a source-listener pair. Table 1 shows the time taken to generate IRs of different lengths and different number of acoustic paths on a laptop with i5-4200M processor and 8 GB of RAM. It can be seen that the time taken remains relatively unchanged across different number of acoustic paths and the depends only on the lengths of the generated IRs.

IR Length (samples)	Computation Time / IR (ms)			
	Number of Acoustic Paths			
	10k	25k	50k	100k
24000	6.0	5.8	6.2	6.1
48000	11.3	11.3	11.4	12.0
96000	23.6	23.4	23.7	24.8

Table 1. Computation time of IRs of different lengths and different number acoustic paths.

2.3 vaRays Auralizer

"Auralization" is the process of sonification of the acoustics in an environment by means of physical and mathematical modelling of the propagation of acoustic energy to induce a sense of immersion in the space [3]. In vaRays, the auralizer subsystem is responsible for producing audio output for a given scene configuration.

The vaRays auralizer uses the *zita-convolver* library by Fons Adriaensen [2] to convolve audio signals from sources with their respective RIRs using the non-uniform partitioned overlap-and-save method (NUPOLS) [11]. The NUPOLS method is a block convolution algorithm where the IR is partitioned into chunks of non-decreasing lengths to provide zero latency and increased computational efficiency.

In dynamically changing scene configurations, the RIRs also change dynamically, requiring live updates to the convolution kernel. Several additions were made to the zita-convolver in order to facilitate real-time IR updates. Smooth transition between IRs during updates is necessary to avoid audible artifacts especially in highly dynamic scenes considering the stochastic nature of the ray tracing algorithm and the absence of path tracking mechanism. Cross-fading between the old and the new IR is a popular method for smooth transition and there are several ways in which this can be accomplished. Some IR update strategies are described in chapter 6 of [12].

Among the ones described, the two main strategies we considered were: 1) the immediate coherent exchange, where all the IR partitions are simultaneously cross-faded over one partition duration, and 2) the successive asynchronous exchange, where the IR partitions are cross-faded successively. After experimentation, we chose the first method since for very long IRs the updates are instantaneous, hence reducing the wait time for all partitions to be swapped. However, the drawback of this method is that if the new IR is much longer than the previous one, the older buffered samples will erroneously be convolved with the new IR and become audible. This can be overcome by a hybrid approach where the immediate coherent exchange strategy is employed when the IR length decreases and successive asynchronous exchange strategy is used when it increases. This will be implemented in vaRays in a future release.

Another important acoustic phenomenon that takes place in dynamic environments with moving sources and listen-

[2] `http://kokkinizita.linuxaudio.org/linuxaudio/index.html`, accessed March 2021.

ers is Doppler shift [13]. In vaRays, this is auralized by the Doppler shift engine.

2.3.1 *Doppler shift engine*

The change in apparent frequency of sound due to relative motion between the source and the listener is known as Doppler shift or Doppler effect. In room acoustic simulators that use the delay-line approach for auralization [4], Doppler shift can be produced by updating delay lengths at audio-rate. However, it is impossible to produce Doppler shift in auralization systems that use the convolution approach. In vaRays, we use a delay-line based Doppler shift engine like in [14], alongside the convolution engine to auralize Doppler shift only for direct paths and ignore the reflected paths. With the addition of a path tracking mechanism, Doppler shift may be produced for persistent reflected paths as well.

Doppler effect depends on the instantaneous relative velocity between a source and a listener, which is usually provided by game physics engines. However, in the absence of a game physics engine, the instantaneous velocities must be computed from positional information. In vaRays, we use a high resolution clock to measure the time difference between successive positional updates for each source-listener pair to estimate the relative velocity.

3. EXAMPLE USAGE

In this section, we discuss two different use cases for vaRays which we have tested:

- Live navigation in a 3D model filled with sound sources, spatialized with vaRays and rendered into binaural audio [3] .
- Live auralization of live acoustical instrument while navigating in geometry displayed inside our 12-meter large dome [4] .

Before describing these use cases in more detail, we introduce the technical pipeline in which vaRays has been integrated with the other tools built at the SAT.

3.1 Technical pipeline

There are three main components in the pipeline:

1. Edition-In-Situ [15] (EiS), a 3D environment for in-situ editing of a virtual scene. We use this tool for visualization and navigation of the environment in real-time.
2. a vaRays application developed in C++ using the vaRays library. This application provides the interface required to modify the virtual environment and perform acoustic simulation in real-time.
3. SATIE [16], a spatial audio scene manager for decoding the ambisonic audio stream in binaural. SATIE is also capable of decoding the ambisonic stream into VBAP for auralization in multi-speaker setups [5].

Fig. 6 shows a block diagram indicating communication between the three components in the pipeline. This pipeline requires an active Jack audio [5] server for routing audio between vaRays and SATIE. The ambisonic order to be used for auralization is set on start-up and a corresponding number of Jack output ports for vaRays and Jack input ports for SATIE are created. Further, a number of JACK output ports are created for SATIE to stream the decoded audio stream. For example, 2 output ports are created when decoding the ambisonic output of vaRays to binaural.

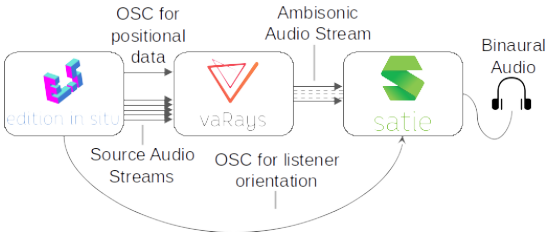


Figure 6. Information flow between different pipeline components.

The acoustic scene in vaRays is updated via OSC messages from EiS. Each time a new source is created, EiS creates a Jack output port with the selected audio source stream and sends an OSC message to vaRays to create a new Jack input port to receive the stream to be convolved with the generated RIR for the source. When a source or the listener moves in the scene, the new positions are updated via OSC and a new ray tracing iteration is triggered in vaRays to recompute the ambisonic IR. The output ambisonic stream from vaRays contains the entire scene audio, i.e., it is a sum of ambisonic streams from all sources in the scene, for a fixed listener orientation. The ambisonic rotation corresponding to the listener orientation is performed by SATIE before the binaural decoding based on Euler angles received directly from EiS.

3.2 Bretez

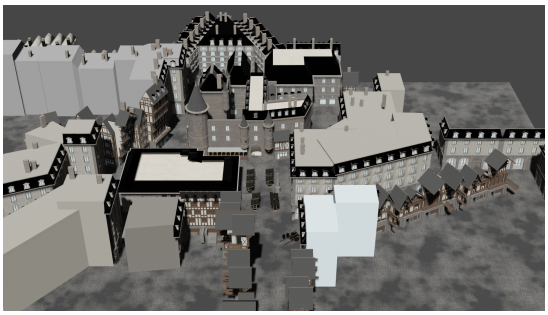
The Bretez project aims at developing immersive editing tools for historians, helping in creating immersive experiences of 18th century Paris [17]. In order to simulate realistic audio in real time, we use the vaRays with the pipeline described in Sec. 3.1. The ray tracing computation time depends on the complexity of the room geometry in the scene. It is advised to use a low-resolution/low-poly version of the room geometry to speed up the ray tracing process while maintaining acoustic simulation quality, thereby reducing the latency between scene changes and the auralized output. We, however, use a high resolution model

[3] The following video, `https://vimeo.com/507255065`, accessed March 2021 shows an example of the Bretez use case.

[4] The following video, `https://vimeo.com/484122810`, accessed March 2021, shows a live auralization of acoustical instruments using vaRays.

[5] `https://github.com/jackaudio/jackaudio.github.com/wiki`, accessed March 2021.

for graphic rendering on EiS for a better visual experience. Fig. 7 shows the models used for graphics and acoustics in this demonstration.



(a) Model used for graphic rendering

(b) Low resolution/low-poly model used for acoustic ray tracing

Figure 7. Two version of the 18th century Paris from the Bretez project. These two versions are aligned and optimized for the two following separate operations: graphic rendering and audio ray tracing.

3.3 Live auralization

We have also used the pipeline described in Sec. 3.1 to create a live auralization experience using vaRays, except for SATIE providing ambisonic order three decoded for our dome speaker system. By using only one source within the scene and using the same coordinates as the listener for it, we create an acoustic experience that corresponds to experiencing the acoustics of a simulated room geometry. We use a mono microphone input as the one source, simulating in real time your own presence in a virtual room. We have tested this by moving the editor in the virtual environment in real time while recording acoustic instruments with the microphone and playing it back in a multichannel audio setup using ambisonic decoding.

4. CONCLUSION AND FUTURE WORK

In this paper, we have described vaRays and its different sub-systems in detail along with some design approaches, challenges and known shortcomings of the library. This library is intended for use with popular game engines and 3D renderers supporting C++ or Python scripting, making an attempt to reduce the threshold of entry into the world of acoustic simulation and facilitate more realistic artistic and immersive experiences.

The choice of Python for a first wrapper around the C++ library is to further simplify integration of vaRays into different software tools, since Python has lately become the de-facto programming language for integrating software building blocks.

Future work will be on the inclusion of transmission of energy through surfaces, diffraction and source directivity in the physical simulation of acoustic propagation, all of which should increase its realism. Alternative simulation methods should be considered for late reverberations as this implies a lot of reflections (hence computations) with our ray-tracing approach.

We are also working in the direction of simulating non-fixed 3D scenes which is mostly a matter of updating the 3D geometry used for ray-tracing, as well as on the creation of multiple datasets comprised of physical measurements of room impulse responses and 3D reconstruction of the recorded location, to allow for validating the employed simulation method.

Lastly, even though the current implementation can handle realtime simulation there is room for many optimizations which should be applied before our method is compared to other methods, like geometrical acoustic solvers for example.

We, at SAT, strongly believe in open-source software since it bolsters scientific and technological evolution by democratizing knowledge. By making vaRays open-source, like the other tools developed at SAT, we hope it will greatly benefit the community of digital artists and acoustic researchers alike.

Acknowledgments

We appreciate the efforts of Julien Wantz, a former Metalab intern who started the vaRays codebase, contributions of Gabriel Downs, Marie-Eve Dumas and Michal Seta to the project, and thank Mylène Pardoen for providing us the Bretez model of 18th Century Paris and sound assets. This project is funded by the Ministère de l'Économie et de L'Innovation du Québec (Canada) under the SAV+R project.

5. REFERENCES

[1] M. Geier, J. Ahrens, and S. Spors, "Object-based audio reproduction and the audio scene description format," *Organized Sound*, vol. 15, pp. 219–227, 10 2010.

[2] L. Savioja and U. P. Svensson, "Overview of geometrical room acoustic modeling techniques," *The Journal of the Acoustical Society of America*, vol. 138, no. 2, pp. 708–730, 2015. [Online]. Available: https://doi.org/10.1121/1.4926438

[3] M. Kleiner, B.-I. Dalenbäck, and P. Svensson, "Auralization-an overview," *J. Audio Eng. Soc*, vol. 41, no. 11, pp. 861–875, 1993. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=6976

[4] D. Poirier-Quinot, B. F. Katz, and M. Noisternig, "EVERTIMS: open source framework for real-time auralization in architectural acoustics and virtual reality,"

in *Proceedings of Digital Audio Effects (DAFx)*, New York, NY, USA, 2017.

[5] N. Bouillot, M. . Seta, Émile Ouellet-Delorme, Z. Settel, and E. Durand, "Rendering of heterogeneous spatial audio scenes," in *Proceedings of the 17th Linux Audio Conference (LAC-19)*, Mar. 2019. [Online]. Available: https://gitlab.com/sat-metalab/metalabbib/raw/master/papers/2019bouillot2.pdf?inline=false

[6] D. Schröder, "Physically based real-time auralization of interactive virtual environments," Ph.D. dissertation, Berlin, 2011, zsfassung in dt. und engl. Sprache. - Druckausgabe: 2011. - Onlineausgabe: 2012; Zugl.: Aachen, Techn. Hochsch., Diss., 2011. [Online]. Available: http://publications.rwth-aachen.de/record/50580

[7] M. Vorländer, "Simulation of the transient and steady-state sound propagation in rooms using a new combined ray-tracing/image-source algorithm," *The Journal of the Acoustical Society of America*, vol. 86, no. 1, pp. 172–178, 1989. [Online]. Available: https://doi.org/10.1121/1.398336

[8] ——, *Auralization: Fundamentals of Acoustics, Modelling, Simulation, Algorithms, and Acoustic Virtual Reality*, 1st ed. Springer-Verlag Berlin Heidelberg, Jun. 2008, no. 1865-0899. [Online]. Available: https://www.springer.com/gp/book/9783540488293

[9] H. E. Bass, L. C. Sutherland, A. J. Zuckerwar, D. T. Blackstock, and D. M. Hester, "Atmospheric absorption of sound: Further developments," *The Journal of the Acoustical Society of America*, vol. 97, no. 1, pp. 680–683, 1995. [Online]. Available: https://doi.org/10.1121/1.412989

[10] F. Harris, E. Venosa, X. Chen, P. Muthyala, and C. Dick, "An extension of the Linkwitz-Riley crossover filters for audio systems and their sampled data implementation," in *2013 20th International Conference on Systems, Signals and Image Processing (IWSSIP)*, Jul. 2013, pp. 175–178, iSSN: 2157-8702.

[11] W. G. Gardner, "Efficient Convolution without Input/Output Delay." Audio Engineering Society, Nov. 1994. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=6335

[12] F. Wefers, M. Vorländer, and L. Savioja, *Partitioned convolution algorithms for real-time auralization*, ser. Aachener Beiträge zur technischen Akustik. Berlin: Logos-Verl, 2015, no. 20.

[13] J. Smith, *Physical Audio Signal Processing, online book, 2010 edition*. W3K Publishing, 2010. [Online]. Available: https://ccrma.stanford.edu/~jos/pasp/pasp.html

[14] C. Schissler and D. Manocha, "Interactive sound propagation and rendering for large multi-source scenes," *ACM Trans. Graph.*, vol. 36, no. 4, Sep. 2016. [Online]. Available: https://doi.org/10.1145/3072959.2943779

[15] F. U. Brien, E. Durand, J. Soria, M. . Seta, and N. Bouillot, "In situ editing (EiS) for fulldomes," in *23rd ACM Symposium on Virtual Reality Software and Technology (VRST)*, Gothenburg, Sweden, Nov. 2017. [Online]. Available: https://gitlab.com/sat-metalab/metalabbib/raw/master/papers/2017brien.pdf?inline=false

[16] N. Bouillot, Z. Settel, and M. Seta, "Satie: a live and scalable 3d audio scene rendering environment for large multi-channel loudspeaker configurations," *New Interfaces for Musical Expression*, 2017.

[17] J. Puget, M. Pardoen, N. Bouillot, E. Durand, M. . Seta, and P. Bastien, "Rapid prototyping of immersive video for popularization of historical knowledge," in *Proceedings of the Thirteenth International Conference on Tangible, Embedded, and Embodied Interaction*, ser. TEI '19. New York, NY, USA: ACM, 2019. [Online]. Available: https://doi.org/10.1145/3294109.3300977

BUILDING NAVIGABLE LISTENING EXPERIENCES BASED ON SPATIAL SOUNDFIELD CAPTURE: THE CASE OF THE ORCHESTRE SYMPHONIQUE DE MONTRÉAL PLAYING BEETHOVEN'S SYMPHONY NO. 6

Zack SETTEL (zsettel@sat.qc.ca) [1,2], **Jean-Yves MUNCH** (jymunch@sat.qc.ca) [1,2],
Gabriel DOWNS (gdowns@sat.qc.ca) [1], and **Nicolas BOUILLOT** (nbouillot@sat.qc.ca) [1]

[1] **Musique6D**, Montreal, Quebec Canada
[2] **Society for arts and technology**, Montreal, Quebec Canada

ABSTRACT

We present our work towards the recording and reproduction of a captured soundfield as a virtual walking experience. In this context, we propose a methodology for recording symphonic concerts that captures both stage and concert hall listening perspectives. The recreation of the experience allows for a virtual walker to explore a plurality of listening perspectives in the venue, and is attended by new challenges in performance capture, mastering and, rendering–all of which stem from the need to maintain a musically balanced listening experience that can adapt to a constantly changing listener position. We present our approach through the creation of an immersive music experience which is deployed for evaluation on two different kinds of audio displays: one with binaural rendering to headphones, and the other that renders to a hybrid loudspeaker system comprising our Satosphere (a large dome with a total of 157 speakers aggregated into 31 channels) for far field audio, and our set of five 12-channel spherical speakers (called Audiodice) for near field audio. The collected data is composed of 64+ audio channel recordings of the Orchestre Symphonique de Montréal conducted by Kent Nagano and a 3D model of the Maison Symphonique de Montréal. This data is described in detail below, and made available with a non-commercial Creative Commons licence.

1. INTRODUCTION

Mainstream uses of game engines and related technological advancements in authoring and rendering 3D object audio scenes have paved the way for a new paradigm in music experience, and by extension, music creation. Using these novel technologies, music can be represented as a spatial arrangement of sound source objects in a virtual 3D audio scene in which a virtual listener is located. The quality of the listening experience, or *listening perspective* as we call it, is determined by the position and orientation, or *6DoF pose* (Degrees of Freedom), of the virtual listener with respect to the surrounding sound source objects in the audio scene. Moving the listener and/or sources within the audio scene yields a change to the listening perspective, and thus allows a listener to focus or *zoom* his/her listening attention on different features of the music. This activity changes the relative intensities among source sounds that comprise a piece of music and thus can eliminate auditory masking effects [1], allowing for otherwise inaccessible features of the music to be revealed. Simply put, the ability to navigate one's listening perspective *within* a musical structure can translate to the ability to hear more, especially when listening to music of greater timbral and/or compositional complexity. Interestingly enough, our informal experiences with the public suggest that this ability tends to benefit trained and untrained listeners alike.

In this paper we describe our novel approach to ensemble recording, mastering, and music listening. What sets our approach apart from traditional methods is the ability to provide a plurality of listening perspectives within the recording space. While microphone placement is key in all approaches, our underlying strategy greatly differs. In traditional classical recording the microphones are strategically placed within the recording space to constitute a *single* listening perspective. Remixing the captured audio streams to constitute other listening perspectives is ineffective because the originally targeted listening perspective is "baked in" to the streams. Conversely, in our approach, a comparatively greater number of microphones is strategically deployed and distributed to capture *zones* of musical interest within the recording space. Subsequently, the set of captured audio streams can be blended to create unique listening perspectives that correspond to *locations* within the physical recording space. Using this technique, audio renderers allow a user to dynamically transition, or *navigate*, his or her listening perspective throughout the captured recording space. In the following discussion we will present our approach and corresponding end-to-end pipeline, *from instrument to ear drum*, which comprises capture, authoring, and rendering stages.

This paper provides a background about sound field capture (Section 2) and discusses our 6DoF capture strategy in a large-scale symphonic music deployment (Section 3). We describe the dataset and how it is distributed for future research (Section 4). Finally, we demonstrate the use of the dataset for prototyping immersive experiences involving live navigation within the captured sound field (Sec-

tion 5). We conclude with a discussion of our results and enumerate our work's future directions (Section 6).

2. BACKGROUND

Human beings moving in space rely on hearing cues that contribute to a spatial understanding and appreciation of our surrounding sound field and space. The simulation of this primal sensation imposes many challenges regarding sound capture and reproduction. One of these challenges is the ability to reproduce navigation in an audio scene along 6 degrees of freedom (6 DoF), i.e. lateral, height and side rotations with translation along the 3 Cartesian axes.

According to Zhang *et al.* [2] spatial audio capture is divided into two main families: binaural capture and sound field capture. Binaural capture, by definition, provides a realistic stereo image in terms of sound depth but suffers from several major problems:

- the captured binaural signal does not allow navigation in the audio scene (0 DoF). Indeed, each microphone corresponds to one ear, not allowing different orientation afterwards when listening.
- the natural filter imposed by the shape of the listener's head and ears contribute significantly to the perception of sound localization. As a result, one must be able to measure these coefficients for each listener [2] if one wants to provide a faithful spatial listening.

Sound field capture using 3D microphones allows sound to be recorded in all directions simultaneously, offering three additional degrees of freedom: lateral, vertical, and side rotation. While this capture method allows the sound field to be captured as a whole, sound sources are localized without being differentiated into independent audio channels. There is no known method today to infer the perspective of a sound scene from a single surround capture, but from another position in space. This is mainly due to the difficulty of isolating the sounds constituting the sound scene, their position, as well as the varying acoustics of complex geometry of the environment [3].

The use of 3D microphone arrays is still quite marginal and is gradually addressing the problems imposed by 6 DoF navigation. For example, [4] improves the performance of sound field capture by reducing the number of microphones needed for audio capture to follow a mobile sound source, such as a teleconference speaker [5]. This research emphasizes the accurate capture of a 2D shot by focusing primarily on the capture of mobile sound sources.

For the production of audio content in Virtual Reality, an English team evaluated the use of several microphone matrix configurations (3 and 6 DoF), with the objective of producing a binaural rendering synchronized with immersive video navigation [6, 7]. More recently, the company ZYLIA [1] has experimented with sound field capture using a 3D microphone array, with the aim of producing a 6-DoF navigation system [8]. This approach

[1] http://www.zylia.co (accessed Dec. 2020)

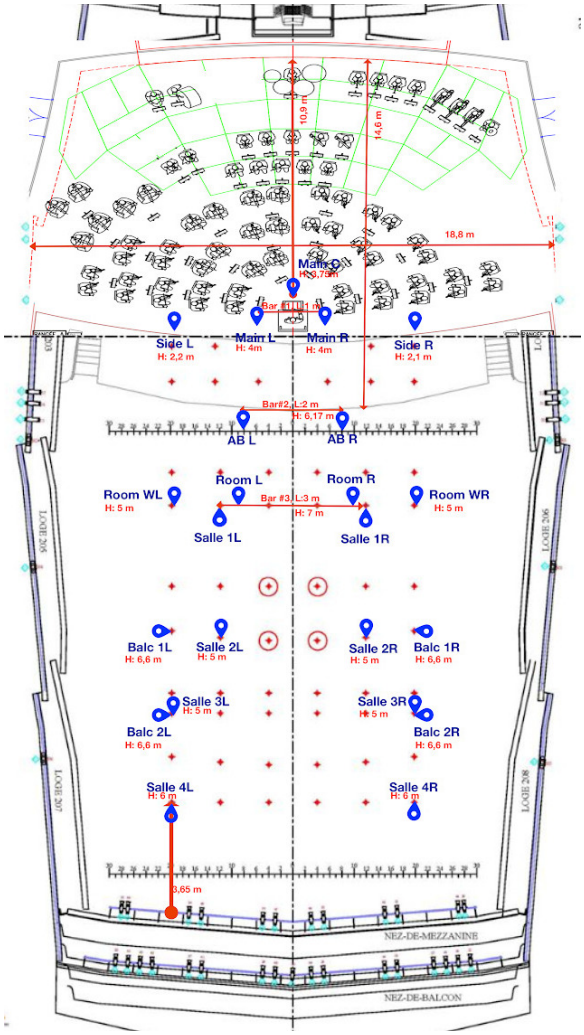


Figure 1. Microphone placement map for room capture

is based on a regular disposition of microphones in the space. Our approach provides more extensive "near-field" navigation where musician-specific locations are captured and provides independent gain control of acoustic (room microphones) and source (stage microphones) energy for wet/dry balancing.

Applications making use of acoustic space modelling using in situ measurements do not usually provide a 6 DoF navigation experience of the simulated space [9]. However, just like the shape and characteristics of sound sources, the reverberation effects of places are determining factors in the final sound effect perceived by the listener [10].

3. 6 DOF CAPTURE

While a conventional stereo master is created for a single listening perspective, a 6DoF master must accommodate a plurality of unique potential listening perspectives that could be rendered for a listener. The captured audio scene

consists of two basic types of sound sources: instrumental and reverberant. The main challenge is to capture a maximum amount of local sonic detail at data rates that can be rendered afterwards in real-time. The microphone set density, types and placement among the instruments or within the concert hall address this challenge.

To capture reverberation in the hall, we deploy 23 microphones, including the main rig, which is composed of three microphones arranged in a Decca tree [11]. These microphones are symmetrically positioned along the length of the venue. All are suspended from the ceiling at increasing heights from front to back. Most of the microphones are omnidirectional with a few exceptions. Room microphone detail is provided in Table 2, while on-stage microphone placement for instrumental capture is illustrated by Fig. 1. Most of the choices for microphone directivity patterns are inspired from state-of-the-art aspects of recording acoustic classic music [12].

3.1 Recording space

The Maison Symphonique de Montreal [2] is the performance venue of the Orchestre Symphonique de Montréal and its conductor, Maestro Kent Nagano. Built in 2011, it was designed by Diamond Schmitt Architects and Ædifica with lead acoustician Bob Essert. The tall and narrow room, based on a traditional shoebox design, is a composition of convex curves of varying size, spreading out the sound and creating a warm, lush timbre. "The walls are large, gently convex panels, designed to spread the sound throughout the hall, to create an intense and immersive sound, drawing the audience into the performance. The scale and curvature of the panels are carefully tailored to refine the frequency spectrum, and therefore the timbre of the sound." [3] The auditorium meets noise criterion N1, in which the background noise level in the hall is not audible to the human ear. The recording control room is close to the stage on the left.

For our recordings, the Orchestre Symphonique de Montréal, conducted by Maestro Kent Nagano, was performing Beethoven's 6th Symphony (Pastorale) on February 18th 2020 with rehearsals on February 16th and 17th. The orchestra is arranged in an antiphonal setup, so the first violins are on the left of the conductor, with the second violins on his right. This is the main difference from the more common Stokowski Shift seating arrangement [13].

3.2 Capture Description

The equipment used for the recording is based on a MADI backbone and Merging Technologies Pyramix DAW. The recording session running on two redundant Window laptops is set @ 96kHz / 24 bits, 62 discrete sources are recorded along with a guide stereo mixdown. All microphone preamps are remote-controlled from the recording booth, so they can be placed close to the sources to which they are connected. The microphones are divided in two groups. Group 1 includes all the point source inputs that are placed on stage, recording local instruments. Group 2 represents the room sources; all microphones are suspended from the ceiling. As the capture took place during a live performance in a packed venue for Maestro Kent Nagano's final concert with the Orchestre Symphonique de Montréal (OSM), we needed to integrate the following constraints:

- No microphone stands or running cables in the space where the audience was seated.
- All room microphones had to be suspended from the ceiling.
- A limited number of cable-pass-thru holes in the ceiling that thus determined microphone positions.
- Limited time to suspend microphones and make adjustments.
- A one-take session, in which a single performance was captured, removing the option to edit in post-production.

The 38 microphones on stage capture isolated instruments or groups of instruments. They are mounted on microphone stands and connected to the microphone preamps through multi-pair cables. Original files are captured as a multi-channel `wav` file with 96kHz sample rate and 24 bit depth. More details are provided by Table 1 and microphone placement on the stage is illustrated by Fig. 2. In the tables, X, Y and Z are expressed in meters, the origin is located at the feet of the conductor, X being directed toward its right (left-right axis), Y toward its front (front-back axis), and Z towards its top (vertical axis).

4. DATASET

As part of our work, we have created a dataset of high quality live recordings of Beethoven's 6th Symphony performed by the OSM. We used these recordings to drive our navigable virtual model of the symphony hall described in Section 5. The dataset consists of 62 `wav` files, one for each microphone placed in the hall. Each file has the same duration, containing a single audio channel that was captured at a specified location in the hall during a live performance of the symphony. The dataset also includes a 3D model of the Maison Symphonique de Montreal, containing a simple representation of the performance space, as well as both groups of microphones deployed in the hall and on stage respectively.

We have made the complete dataset freely available for download at `https://archive.org/details/savr_audio_dataset` in the hopes that others will be able to use it for further audio research. These recordings are licensed under a Creative Commons Attribution Non-Commercial ShareAlike 4.0 International License.

[2] `https://www.osm.ca/en/the-hall/`, accessed in March 2021.

[3] Essert, B. (2011) Acoustics Design – La Maison Symphonique, Montreal in `https://www.soundspacevision.com/`, accessed Feb. 2021.

Group	#	label	Microphone	Preamp	X	Y	Z	Filename	Directivity
1st violin	24	V1.1	Neumann KM 143	Grace Design m108 #4	-1.43	1.08	1.5	024_V1_1	cardioid
	25	V1.2	Neumann KM 140	Grace Design m108 #4	-2.91	1.36	1.5	025_V1_2	cardioid
	26	V1.3	Schoeps CMC621	Grace Design m108 #4	-4.26	1.49	1.5	026_V1_3	between omni and cardioid
	27	V1.4	Schoeps CMC621	Grace Design m108 #4	-5.56	1.74	1.5	027_V1_4	between omni and cardioid
	28	V1.5	Neumann KM 140	Grace Design m108 #4	-7.05	2.06	1.5	028_V1_5	cardioid
2nd violin	29	V2.1	Neuman KM 143	Grace Design m108 #4	1.2	1.04	1.5	029_V2_1	between omni and cardioid
	30	V2.2	Neuman KM 140	Grace Design m108 #4	2.74	1.32	1.5	030_V2_2	cardioid
	31	V2.3	Neuman KM 140	Grace Design m108 #4	3.85	1.53	1.5	031_V2_3	cardioid
	32	V2.4	Neuman KM 140	RME Octamic XTC #1	5.39	1.88	1.5	032_V2_4	cardioid
	33	V2.5	Neuman KM 140	RME Octamic XTC #1	6.86	2.26	1.5	033_V2_5	cardioid
Violas	34	Va.1	Sennheiser MKH 8040	RME Octamic XTC #1	0.64	2.3	1.5	034_Va_1	cardioid
	35	Va.2	Sennheiser MKH 8040	RME Octamic XTC #1	2.35	3.27	1.5	035_Va_2	cardioid
	36	Va.3	Schoeps CMC622	RME Octamic XTC #1	0.91	4.23	1.5	036_Va_3	between omni and cardioid
	37	Va.4	Schoeps CMC622	RME Octamic XTC #1	2.83	4.72	1.5	037_Va_4	between omni and cardioid
	38	Va.5	Neuman KM 140	RME Octamic XTC #1	1.26	5.59	1.5	038_Va_5	cardioid
Cellos	39	Vc.1	Coles 4038	RME Octamic XTC #1	-0.84	2.35	0.8	039_Vc_1	figure eight
	40	Vc.2	Sennheiser MKH 800	Grace Design m108 #5	-2.54	2.82	0.8	040_Vc_2	cardioid
	41	Vc.3	Sennheiser MKH 800	Grace Design m108 #5	-0.93	4.09	0.8	041_Vc_3	cardioid
	42	Vc.4	Sennheiser MKH 800	Grace Design m108 #5	-2.86	4.67	0.8	042_Vc_4	cardioid
	43	Vc.5	Sennheiser MKH 800	Grace Design m108 #5	-1.08	5.55	0.8	043_Vc_5	cardioid
Basses	44	Vb.1	Coles 4038	Grace Design m108 #5	-5.38	5.42	0.8	044_Vb_1	figure eight
	45	Vb.2	Neumann TLM 103	Grace Design m108 #5	-6.98	3.97	0.8	045_Vb_2	cardioid
	46	Vb.3	Sennheiser MK 8020	Grace Design m108 #5	-7.85	5.52	0.8	046_Vb_3	omni
Woodwinds	47	Fl.2	Neumann KM 84	Grace Design m108 #5	-1.49	6.93	1.3	047_Fl_2	cardioid
	48	Fl.1	Neumann KM 84	RME Octamic XTC #2	-0.68	6.99	1.3	048_Fl_1	cardioid
Oboes	49	Ob.1	Neumann KM 84	RME Octamic XTC #2	0.26	6.91	1.3	049_Ob_1	cardioid
	50	Ob.1	Neumann KM 84	RME Octamic XTC #2	1.14	6.88	1.3	050_Ob_2	cardioid
Clarinets	51	Kl.2	Neumann KM 84	RME Octamic XTC #2	-1.55	8.26	0.8	051_Clar_2	cardioid
	52	Kl.1	Neumann KM 140	RME Octamic XTC #2	-0.79	8.317	0.8	052_Clar_1	cardioid
Bassoons	53	Fg.1	Neumann KM 85	RME Octamic XTC #2	0.87	8.439	1	053_Bsn_1	cardioid
	54	Fg.2	Neumann KM 140	RME Octamic XTC #2	1.736	8.167	1	054_Bsn_2	cardioid
Horns	55	Horn.1	Neumann KM 84	RME Octamic XTC #2	-4.34	8.85	0.8	055_Hrn_1	cardioid
	56	Horn.2	Neumann KM 84	Millennia HV-3R	-5.11	8.32	0.8	056_Hrn_2	cardioid
Trumpets	57	Tpt.1	AEA R84	Millennia HV-3R	2.47	10.65	1	057_Trp_1	figure eight
	58	Tpt.2	AEA R84	Millennia HV-3R	3.65	10.36	1	058_Trp_2	figure eight
Trombones	59	Tbn.1	Neumann TLM 67	Millennia HV-3R	4.92	9.84	1	059_Tbn_1	cardioid
	60	Tbn.2	Neumann TLM 67	Millennia HV-3R	6.03	8.89	1	060_Tbn_2	cardioid
Timpani	61	Tim.1	AKG c414	Millennia HV-3R	-0.51	11.13	1.9	061_Timb_1	cardioid
	62	Tim.2	AKG c414	Millennia HV-3R	0.4	11.19	1.9	062_Timb_2	cardioid

Table 1. Specification of microphones used for the audio capture of instruments.

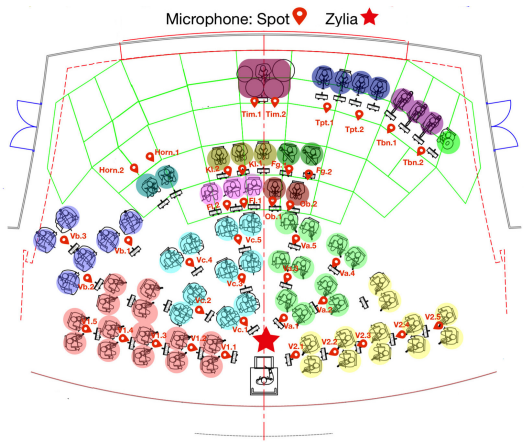


Figure 2. Microphone placement map for instrument capture.

5. NAVIGATION IN THE ORCHESTRA

5.1 Concepts and Strategies

The ability to navigate a listening perspective *anywhere* within a virtual audio scene is at the core of our approach, and greatly differs from methods used in conventional recording and mastering, which seeks to provide one *single* pre-determined listening perspective.

Our work combines theoretical and empirical approaches to the authoring and rendering of navigable symphonic music listening experiences. This includes a musical orientation, tempered by years of first-hand experience in orchestral sound recording, mastering and music composition. In working with the presented orchestral music, the goal has been to create a navigable music scene with the following essential properties:

1. The scene is balanced among captured channels from any listening position.

2. The scene provides a high degree of differentiation and detail when zoomed on particular instrument zones.

3. The scene provides a natural sounding stereo image whose width changes appropriately with distance.

4. The scene provides, at any distance, a natural sounding balance of reverberant and instrumental sound sources.

#	label	Microphone	Preamp	Hole#	X	Y	Z	Height (m)	Filename	Directivity
1	Main L	DPA 4006	Grace Design m108 #1	23	-1	0.04	4	4.00	001_L	omni
2	Main R	DPA 4006	Grace Design m108 #1	22	1	0.04	4	4.00	002_R	omni
3	Main C	DPA 4006	Grace Design m108 #1	19/18	0	1.37	3.45	3.75	003_C	omni
4	Side L	Sennheiser MKH8020	Grace Design m108 #1	23	-5.9	0.04	2.2	2.23	004_Side	omni
5	Side R	Sennheiser MKH8020	Grace Design m108 #1	22	5.9	0.04	2.1	2.10	005_Side	omni
6	AB L	DPA 4006	Grace Design m108 #1	28	-1	-4.44	6.17	6.17	006_AB	omni
7	AB R	DPA 4006	Grace Design m108 #1	27	1	-4.44	6.17	6.17	007_AB	omni
8	Room L	Sennheiser MKH8020	Grace Design m108 #1	34	-1.3	-8.96	7	7.00	008_Room	omni
9	Room R	Sennheiser MKH8020	Grace Design m108 #2	33	1.3	-8.96	7	7.00	009_Room	omni
10	Room WL	DPA 4006	Grace Design m108 #2	36	-5.9	-8.96	5	5.00	010_Room_W	omni
11	Room WR	DPA 4006	Grace Design m108 #2	31	5.9	-8.96	5	5.00	011_Room_W	omni
12	Salle 1L	Schoeps CMC641	Grace Design m108 #2	34	-1.6	-8.96	7	7.00	012_Salle1	hypercardioid
13	Salle 1R	Schoeps CMC641	Grace Design m108 #2	33	1.6	-8.96	7	7.00	013_Salle1	hypercardioid
14	Salle 2L	Sennheiser MKH8020	Grace Design m108 #2	47	3.6	-14.11	5	5.00	014_Salle2	omni
15	Salle 2R	Sennheiser MKH8020	Grace Design m108 #2	44	-3.76	-14.11	5	5.00	015_Salle2	omni
16	Salle 3L	Sennheiser MKH8020	Grace Design m108 #2	54	-5.88	-18.04	5	5.00	016_Salle3	omni
17	Salle 3R	Sennheiser MKH8020	Grace Design m108 #3	49	5.88	-18.04	5	5.00	017_Salle3	omni
18	Salle 4L	Schoeps CMC641	Grace Design m108 #3	54	-6	-22.27	6	6.00	018_Salle4	hypercardioid
19	Salle 4 R	Schoeps CMC641	Grace Design m108 #3	49	6	-22.27	6	6.00	019_Salle4	hypercardioid
20	Balc 1L	Sennheiser MKH8020	Grace Design m108 #3	48	-5.89	-14.11	5	6.60	020_BalcL	omni
21	Balc 2L	Sennheiser MKH8020	Grace Design m108 #3	54	-6.52	-18.04	6.6	6.60	021_BalcL	omni
22	Balc 1R	Sennheiser MKH8020	Grace Design m108 #3	43	6.01	-14.11	5	6.60	022_BalcR	omni
23	Balc 2R	Sennheiser MKH8020	Grace Design m108 #3	49	6.52	-18.04	6.6	6.60	y023_BalcR	omni

Table 2. Specification of microphones used for the audio room capture.

The biggest challenge has been to combine and balance *all* the above in *one single* audio scene. Ideally, at one moment, the listening perspective can be navigated to a central location in say, the 7th row, in order to deliver a well-blended listening experience akin to those targeted by conventional concert recordings. Then at another moment, the perspective can be smoothly navigated to a different location *within* the orchestra to reveal fine sonic detail among the woodwinds. Via strategic capture and rendering techniques, we address the challenges that accompany listening perspective navigation.

5.2 Mixing by Arrangement

Our virtual audio scene is a collection of reverberant and instrumental sound sources, each emitting audio that was captured by a particular microphone. As shown earlier, twenty-eight microphones were suspended from the ceiling to capture reverberation in the concert hall shown in Fig. 1, while thirty-four microphones were positioned on the stage to capture the direct instrumental sounds shown in Fig. 2. The location of each source in the 3D audio scene corresponds to the location of a specific microphone in the real concert hall. Thus, the arrangement of the sound sources in the virtual scene *exactly* corresponds to the microphone arrangement in the recording session as shown in Fig. 3. The quality of the output mix is determined by the virtual listener's position and angle, or 6DoF pose, in the virtual audio scene, with respect to all the scene's sound sources.

5.3 Rendering the Listening Perspective

The listening perspective consists of a weighted sum of processed outputs from all the scene's audio sources. The processing of each audio source is based on the difference between that source's 6DoF pose and the 6DoF pose of the listener. Any time a listener or source moves, the perspective is updated, and low level parameters are computed (azimuth, elevation, gain, delay and spread) based on the following geometrical information: i) The distance between the listener and the source, ii) The angular position of the listener relative to the source, and iii) The incidence of the listener relative to the source.

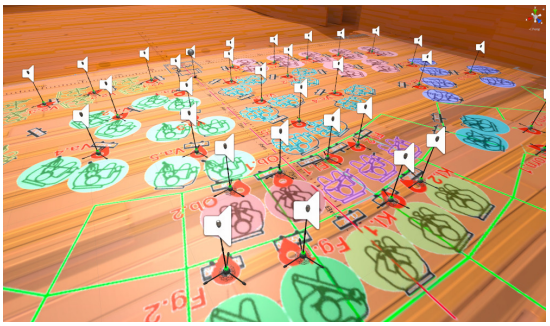


Figure 3. Virtual audio sources positioned *one to one* according to the microphone placement map

Using these relations, we dynamically compute values for each source mixing parameter below: i) Attenuation, using distance and incidence, ii) Near Field Processing, using distance, iii) Filtering using distance and incidence, iv) Spread using distance, and v) Delay or *Doppler* using distance.

Our renderer offers the ability to describe distance/value functions for these parameters, thus allowing us to define custom attenuation-by-distance curves, etc. In addition to the dynamic parameters above, the following static mixing parameters can be adjusted independently for each source: i) Attenuation Trim (db), ii) Doppler Effect (percent), iii) Near Field Effect (percent), and iv) Near Field Radius (meters)

5.4 6DoF Mastering

Unlike a conventional non-navigable master, which is created for a single listening perspective, a 6DoF master must accommodate a plurality of different listening perspectives as shown in Fig. 5. Whereas a conventional master, such

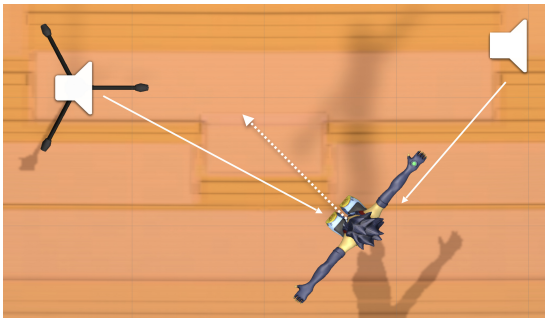


Figure 4. Source audio processing based on relative distances and incidences to listener

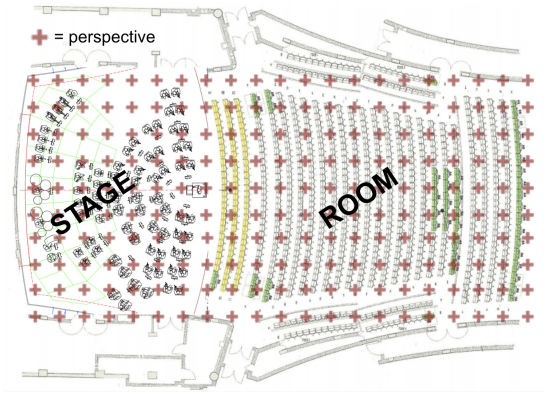


Figure 5. Mastering to accommodate a plurality of listening perspectives

as stereo or surround, is a rendered piece of audio content, the 6DoF master is a *renderable* 3D virtual audio scene. To a certain extent, our workflow resembles a game development environment. In order to create navigable audio scenes, we have developed custom workflows that help us achieve an optimized balance of the essential properties mentioned earlier. To achieve this balance, our workflow allows us to carefully adjust or *tune* both dynamic and static parameters, shown above, for all the scene's reverberant and instrumental sound sources.

Except for the Doppler/delay parameters, which we minimize for musical applications, our parameter adjustments determine the following attributes of the mix:

1. *Wet to Dry* Ratio, which in our case, refers to the balance between reverberant sources and instrumental sources.

2. Near Field Dynamics [14], that determine the quality and amount of *sonic zoom* that occurs when listening close to instrumental sources.

3. Spatial Localization Intensity, the degree to which, a given source is perceived to originate from a singular location in the space surrounding the listener.

We begin the mastering process by pre-establishing, in musical terms, overall ranges for the attributes above as the perspective is moved variously throughout the entire audio scene. We then go about adjusting the mixing parameters in order to achieve a reasonable balance of essential properties across a wide range of potential listening perspective locations. Furthermore, we refer to this activity as *tuning the audio scene*. Based on the type of source sound, reverberant or instrumental, a source's mixing parameter values will be tuned to augment or reduce the degrees of spatial localization, near-field effect, attenuation by distance, etc. To make this authoring process easier and more efficient, we have developed mixing operations such as grouping, for subsets of reverberant sources, and for instrumental sources. In addition to parameter changes, these groups also allow us to mute and offset the gains of their respective sources, as we tune our mixes.

In conventional mastering, the final outcome is the result of iterations of adjustment and evaluation on different loudspeaker configurations. The same holds true for our 6DoF mastering process. To our great surprise, the resulting 6DoF master audio scene yielded the same high-quality music experience when rendered for stereo headphones, and for the 31 channel Satosphere dome, alike. No "retuning" of the audio scene was required when changing among output rendering formats, even when they differed greatly. Even more surprising, when rendering on the hybrid audio system combining the Satosphere and five Audiodice [4] spherical speakers, for far and near field display respectively, the audio scene still did not require adjustment [5].

To date, we have applied our workflow to create unique 6DoF masters for three different symphonic works: Beethoven's Pastoral, Beethoven's Seventh, and Pascal Dusapin's "Wave". In the process, we have observed that reusing an audio scene with parameters adjusted for one piece of music, such as the Beethoven Pastoral, when rendering a different work, such as the Dusapin's "Wave", yielded poorer listening experiences in terms of the essential properties mentioned earlier. As with conventional masters, if not even more so, it would appear that 6DoF masters need to be uniquely crafted to each piece, based on possible variations among sound sources locations and specific nature of the compositions.

5.5 Platform

5.5.1 Rendering

The rendering stage of our pipeline shown in Fig. 6 incorporates both the Unity and SuperCollider environments, each of which has been extended to render the music scene in tandem [6]. Running in SuperCollider, the SATIE [15] library, developed by the SAT Metalab, provides an extremely efficient realtime dynamic audio processing environment, and offers a wide range of extendable output format configurations. The inputs to SATIE consist of raw au-

[4] See the Audiodice prototype building at `https://vimeo.com/519938720`. Accessed March 2021.

[5] See our demonstration video at `https://vimeo.com/519940468`. Accessed March 2021.

[6] The code repository is published under the GPL licence and is available here : `https://gitlab.com/sat-metalab/6dmastering4savr`

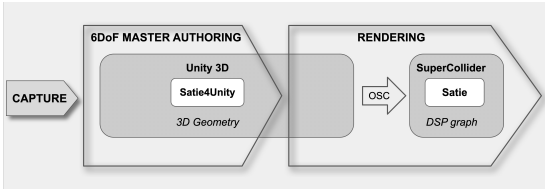


Figure 6. Distributed authoring and rendering pipeline

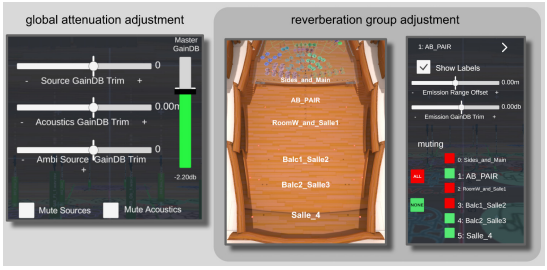


Figure 7. Authoring tools for 6DoF mastering

dio streams and asynchronous rendering messages. SATIE outputs rendered audio streams for one or more listeners, in one or more audio output formats, such as binaural stereo, high-order ambisonics, or the custom 31-channel Satosphere Dome format. SATIE can also simultaneously render near and far field perspectives, as separate audio output streams, which can be useful when using hybrid audio displays [16].

In our pipeline, SATIE functions as an audio processor, computing a node-based DSP graph of sources and syncs. Input messages to SATIE mainly comprise source-specific lists of DSP update parameters for gain, delay and filtering. The messages are sent asynchronously over IP by a library running in the Unity Environment called Satie4Unity, also developed by the SAT Metalab. Satie4Unity provides control for update message density, which we typically limit to no more than 10 to 20 times per second, per source node. In SATIE, the received update message parameters are smoothed.

The input to Satie4Unity is the 3D audio scene graph, or "6DoF master" referred to above, which contains one or more listeners and a spatial arrangement of reverberation and instrumental source objects. Basically, Satie4Unity converts changes to 3D source and listener geometry into batches of source-listener DSP parameter updates, which are transmitted as OSC messages to SATIE.

5.5.2 Authoring

Satie4Unity also extends Unity's native authoring UI, and thus allows us to build tools that integrate existing UI features for editing curves or adjusting parameters, shown in Fig. 7. As our Authoring/Rendering pipeline runs in realtime, these tools serve our requirement to instantly hear adjustments we make to the audio scene during the mastering process.

6. DISCUSSION AND FUTURE WORKS

The approach we describe here provides general guidelines for building navigable listening experiences, and more particularly about the working pipeline strategy. The division of the capture in *zones* (stage vs. hall) has provided expected results in terms of 6DoF mastering flexibility when fine-tuning the 6DoF navigation for different recorded pieces. Additionally, our strategy takes advantage of conventional capturing equipment, techniques and expertise, permitting for the use of particular microphones with specific patterns and preamps.

Our work to date has led us to consider several improvements and promising new deployments to investigate :

1. Use of ambisonic microphones to capture reverberation in the acoustic space. Ambisonic microphones will provide extra directionality and may improve subtle variations in reverberation

2. Add a rendering effect to enhance the degree of spatial encompassment at different distances from a cluster of instrumental source sounds.

3. Use of *outside in* microphone arrays to capture and then render a single source from multiple surrounding angles.

4. Extension of near field rendering techniques for sources captured by *outside in* microphone arrays, and rendered for the Audiodice spherical audio display system.

5. Implementation of geometry-based acoustic modelling for live navigable reverberation rendering with live ray tracing [17].

6. The use of binaural recordings, captured by an itinerant listener during the recording session, for comparative evaluations of rendered listening experiences.

7. Extension of the dataset with directional impulse responses measures of the Maison Symphonique de Montréal

7. CONCLUSION

Indeed, the ability to navigate complex forms of music, such as symphonic works, modern or otherwise, can yield a particularly compelling listening experience in terms of one's ability to recognize and differentiate timbral and structural elements in the music. We are convinced that our efforts have been effective in developing a pipeline that can deliver navigable music listening experiences of high musical quality. We have furthered our understanding of microphone deployment density and placement. In the authoring of the three previously mentioned 6DoF masters, we have addressed the need to potentially render *all* listening perspectives, and created workflows with musically prioritized steps and operations to simplify the task and improve the result. Finally, we have been able to evaluate the effectiveness of music scene organization in terms of

reverberation and instrumental sources. While we are excited by the results we obtained, we recognize that there is much work still to be done, including potential to extend the captured navigable space to improve listening quality at elevated locations in the audio scene. Finally, through our work with real-time interactive experience rendering on a large symphonic scale, we have achieved a difficult balance between data density and musical experience quality.

Acknowledgments

We would like to thank the Orchestre Symphonique de Montréal for its generosity during the research process, and especially for the opportunity to record their performances. Thanks also to Carl Talbot and his team for their significant contribution to the capture and microphone positioning in the Maison Symphonique de Montréal. Thanks to Luc Martinez for his support during the project ideation. Finally, we thank the Ministère de l'Économie et de l'Innovation du Québec for making this project possible.

8. REFERENCES

[1] S. McAdams and A. Bregman, "Hearing musical streams," *Computer Music Journal*, vol. 3, no. 4, pp. 26–60, 1979. [Online]. Available: http://www.jstor.org/stable/4617866

[2] W. Zhang, P. N. Samarasinghe, H. Chen, and T. D. Abhayapala, "Surround by sound: A review of spatial audio recording and reproduction," *Applied Sciences*, vol. 7, no. 5, 2017. [Online]. Available: http://www.mdpi.com/2076-3417/7/5/532

[3] J.-H. Wang, A. Qu, T. R. Langlois, and D. L. James, "Toward wave-based sound synthesis for computer animation," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 109:1–109:16, Jul. 2018. [Online]. Available: http://doi.acm.org/10.1145/3197517.3201318

[4] P. Samarasinghe, T. Abhayapala, and M. Poletti, "Wavefield analysis over large areas using distributed higher order microphones," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 3, pp. 647–658, March 2014.

[5] H. F. Silverman, W. R. Patterson, and J. L. Flanagan, "The huge microphone array," *IEEE Concurrency*, vol. 6, no. 4, pp. 36–46, Oct 1998.

[6] D. Rivas Méndez, C. Armstrong, J. Stubbs, M. Stiles, and G. Kearney, "Practical recording techniques for music production with six-degrees of freedom virtual reality," in *Audio Engineering Society Convention 145*, Oct 2018. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=19729

[7] H. Riaz, M. Stiles, C. Armstrong, A. Chadwick, H. Lee, and G. Kearney, "Multichannel microphone array recording for popular music production in virtual reality," *Audio Engineering Society, Convention e-Brief*, 10 2017.

[8] E. Patricio, A. Ruminski, A. Kuklasinski, L. Januszkiewicz, and T. Zernicki, "Toward six degrees of freedom audio recording and playback using multiple ambisonics sound fields," in *Audio Engineering Society Convention 146*, Mar 2019. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=20274

[9] S. Tervo and A. Politis, "Direction of arrival estimation of reflections from room impulse responses using a spherical microphone array," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 10, pp. 1539–1551, 2015.

[10] B. F. G. Katz and D. Poirier-Quinot, "Archéologie acoustique par reconstruction virtuelle pour l'analyse in situ," *Culture et recherche*, no. 141, pp. 25–26, Sep. 2020. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02940162

[11] J. Borwick, *Sound Recording Practice*, T. Tryggvason, Ed. London: Oxford University Press, 1976.

[12] C. Haigh, J. Dunkerley, and M. Rogers, *Classical Recording: A Practical Guide in the Decca Tradition (1st ed.)*. London: Focal Press, Oct. 2020, https://doi.org/10.4324/9780429316852.

[13] J. Meyer, *Seating Arrangement in the Concert Hall*. New York, NY: Springer New York, 2009, pp. 263–346. [Online]. Available: https://doi.org/10.1007/978-0-387-09517-2_7

[14] D. R. Moore and A. J. King, "Auditory perception: The near and far of sound localization," *Current Biology*, vol. 9, no. 10, pp. R361 – R363, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0960982299802279

[15] N. Bouillot, M. . Seta, Émile Ouellet-Delorme, Z. Settel, and E. Durand, "Rendering of heterogeneous spatial audio scenes," in *Proceedings of the 17th Linux Audio Conference (LAC-19)*, Mar. 2019. [Online]. Available: https://gitlab.com/sat-metalab/metalabbib/raw/master/papers/2019bouillot2.pdf?inline=false

[16] Z. Settel, P. Otto, M. . Seta, and N. Bouillot, "Dual rendering of virtual audio scenes for far-field surround multi-channel and near-field binaural audio displays," in *16th Biennial Symposium on Arts and Technology*, Ammerman Center for Arts and Technology at Connecticut College, New London, Feb. 2018, 5 pages. [Online]. Available: https://gitlab.com/sat-metalab/metalabbib/raw/master/papers/2018settel.pdf?inline=false

[17] Émile Ouellet-Delorme, H. Venkatesan, E. Durand, and N. Bouillot, "Live ray tracing and auralization of 3d audio scene with vaRays (submitted)," in *18th Sound and Music Computing Conference (SMC)*, 2021.

USER HRTF SELECTION FOR 3D AUDITORY MIXED REALITY

Rishi SHUKLA[1], **Rebecca STEWART**[2], and **Mark SANDLER**[1]

[1] *Centre for Digital Music*, **Queen Mary University of London**, London, UK
[2] *Dyson School of Design Engineering* , **Imperial College London**, London, UK

ABSTRACT

We introduce a novel approach for personalisation of an efficient 3D binaural rendering system designed for mobile, *auditory mixed reality* use cases. A head-related transfer function (HRTF) ranking method is outlined for users of real-time, interactive sound and music applications. Twenty participants tested the approach and its impact on their capacity to locate a continuous musical sound rendered in varying 3D positions. Analysis of HRTF rankings across three separate sessions reveal encouraging levels of reliability amongst some participants. Patterns of interaction show a significant benefit to horizontal precision that results from the selection process. In contrast, length of system exposure (rather than HRTF preference) demonstrates a significant degree of improvement to aspects of vertical perception and overall speed of response, with no detriment to horizontal accuracy. These findings provide an initial basis from which to consider priorities in the design of audio-only immersive applications and accompanying methods for effective user controlled personalisation.

1. INTRODUCTION

Since the 1990s, sound computing research has explored spatially located interactive audio within real world contexts. The principles and applications of binaural synthesis (i.e. 360° sound simulation over headphones) were given prominent discussion in Begault's major work on 3D audio for multimedia [1]. Bederson's virtual tour guide presented an early design for personalised real-time audio intervention to augment physical environments [2]. Possibilities for blending acoustic and digitally situated sound were later investigated with closed-ear headphones (Cohen's experiment with artificially spatialised sound sources [3]) and wearable speakers (Sawnhey's Nomadic Radio [4]).

Mobile computing has brought renewed focus on personalised interactive sound design that supplants or supplements environmental acoustics, for example through multilayered spatial auditory display to encourage exploration of art exhibitions [5]. 'Hearable' headsets have further accelerated development. These devices feature integrated orientation and often positional sensing to enable immersive audio spatialisation. Direct or electronically assisted openness to environmental sound is also enabled in some instances. Recent evaluation of these devices has highlighted potential affordances and some initial interaction design recommendations. It has also introduced new working definitions to distinguish the variety of hardware features and modes of experience that coexist in *auditory mixed reality* (AMR), a term that 'encapsulates any auditory VR and AR experiences' (i.e. forms of sonic virtual or augmented reality) [6]. These technologies have recently prompted structured experimentation with creative social gameplay [7], as well as direct industry research addressing how audio virtual reality might be applied to music discovery activity [8].

Research into personalisation of spatial rendering for these audio-only contexts is limited. The potential benefits to user orientation and sense of immersion from incorporating HRTF personalisation have been highlighted in previous spatial audio experience studies as an area requiring structured investigation [5, 9]. This paper presents and evaluates a method for user selection of preferred HRTF sets via a custom binaural rendering system for mobile, interactive spatial audio. Section 2 discusses the limitations of prior work on HRTF selection for end-user audio-only contexts and specific considerations therein. Section 3 outlines the method conceived for 3D HRTF comparison and the experimental protocol for measuring its performance. Results of the evaluation are presented in section 4, followed by discussion of outcomes and summary conclusions in sections 5 and 6.

2. BACKGROUND

HRTF sets are location dependent filters that simulate spatialisation of sources around a listener. HRTF sets comprise multiple head-related impulse responses (HRIRs), which are measured within the left and right ear of a human or dummy head, using an excitation source placed at incremental surrounding positions. The efficacy of HRTF processing is dependent on both the density of the measurements, and correlation between the HRIRs and features of the listener's morphology that affect spatial perception (chiefly the shape of their head, pinnae and upper body) [1, 10]. Use of a generic or poorly matched HRTF set is liable to impact sound localisation accuracy, discrimination between sources in front/behind and above/below, externalisation (i.e. sense that a sound is emanating from outside the lister's head), and tonal clarity.

A growing body of research demonstrates the efficacy of parametric methods for either selecting or simulating best-

fitting HRTF sets according to user morphology [11–14]. However, precise and reliable acquisition of anthropometric head, ear and torso features in the case of mobile system end-users is a nontrivial challenge. Selection of a best-fitting HRTF set from a database of alternatives is another established strategy for customising binaural rendering [15, 16]. However, the latter approach – which is the area of concern for the remainder of this paper – also raises specific design problems for the context of AMR.

2.1 Working criteria for end user HRTF selection

The authors have previously discussed the difficulties of applying established HRTF evaluation methodologies within an end user system [17]. Objective approaches require subjects to repeatedly identify perceived locations of sources placed in various spatial positions, for alternate HRTF sets. This is by its nature time-consuming and typically relies on a laboratory-style format and rapid serial responses to short test signal stimuli [18–20]. Subjective approaches, by contrast, ask users to rate the effect of alternate HRTF sets using separate criteria, such as consistency in motion/trajectory, hemispherical distinction or degree of externalisation [21, 22]. Evidence suggests that repeatability of such qualitative judgements is contingent on listener expertise [23–26].

A design for HRTF selection in 2D was previously devised using interactive holistic A/B comparison, with recorded music as the stimulus signal. The approach was evaluated against these four criteria with encouraging outcomes [17].

- *Reliability* – clear and consistent selection outcomes
- *Validity* – spatial fidelity that is fit for purpose
- *Usability* – of potential benefit to any end user
- *Efficiency* – a duration acceptable for the use case

2.2 Considerations for 3D audio-only mobile contexts

Adapting the 2D holistic approach for 3D judgement of HRTFs in mobile use cases presents three challenges:

1. Thorough comparative judgement of 3D involves a much wider range of spatial positions and trajectories, all of which must be adequately explored to make valid selections.
2. Mobile and 'hearable' devices have either restricted or no visual display, meaning that judgements should be made without reference to dynamic graphics or complex interfaces that would be necessary for audiovisual interaction.
3. A clear indication of selection repeatability is necessary in light of the above complexities.

This research deploys a virtual vector base amplitude panning (VBAP) [27] system developed on an open-source embedded Linux platform. The approach uses eight HRIR pairs to simulate a sparse, pseudo-spherical virtual loudspeaker array. Individual sound sources are positioned via VBAP processing prior to binaural encoding. Five virtual speakers are located on the horizontal plane (0° elevation) at 0°, -60°, 60°, 120° and -120°. Three further speakers are placed around the median plane (0° azimuth) at 90° (directly above), -45° (below front) and -135° (below back). VBAP has been shown to render with favourable levels of spatial and tonal consistency in this virtual configuration [28] and more generically [29]. However, specific limitations to vertical cue representation in a sparse array layout, such as that defined above, are also well known [30].

3. METHODOLOGY

A selection method was devised and evaluated against the four criteria in section 2.1. Participants undertook three identical study sessions (to assess *reliability*). In each session, part one comprised of the HRTF selection procedure (to test *usability* and *efficiency*) and part two consisted of a follow-up objective localisation task (to measure *validity*).

3.1 Participation

Twenty-one participants (aged 25-45, 6 female and 15 male) were recruited on to the study, which was approved by the Queen Mary University of London (QMUL) Ethics Committee (reference 2038). Each session was approximately 45 minutes, with a minimum of 48 hours between sittings. Participants received £20 in compensation for their time at the end of their third session. All participants were recruited via an open call to staff and doctoral students across QMUL's School of Electronic Engineering and Computer Science. No hearing impairments were declared, other than from two participants who reported occasional and slight tinnitus that neither regarded as pronounced. A questionnaire was used to collect information on musical training, headphone listening habits and prior exposure to binaural audio.

Participants wore a battery powered embedded device running the rendering, study journey and response logging software. They used an iOS device to submit responses and progress through the study. Sennheiser HD650 headphones (without any equalisation for binaural synthesis applied) were secured to participants with an elasticated hairband. The head-tracking sensor was mounted on top of the headphones to counter-rotate the sound scene and take head position readings at 1° angular resolution. Head-tracking was only enabled in part two of the study.

3.2 Part one: HRTF selection

A comprehensive tournament structure used 21 pairwise comparisons between the optimised shortlist of seven human-measured HRTF sets from the LISTEN database [31] identified in [32]. A notable previous investigation into HRTF selection repeatability used stimuli with fixed trajectories that were not responsive to head-tracking [23]. The test pursued here followed a similar approach, but used content derived from recorded music, rather than test tone signals. The comparison stimulus was compiled from excerpts of an anechoic recording of Haydn's Trumpet Concerto in Eb performed on unaccompanied cornet [33].

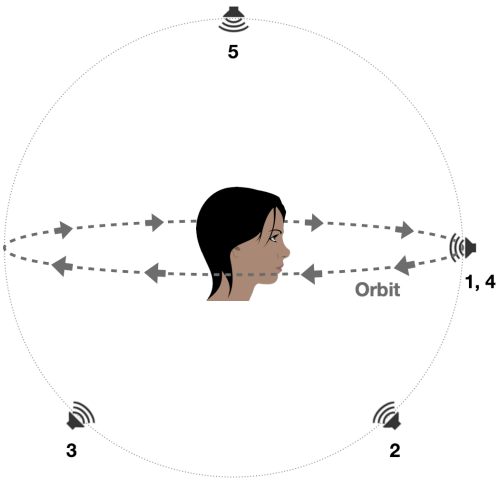


Figure 1. Trajectory for virtual VBAP HRTF comparison

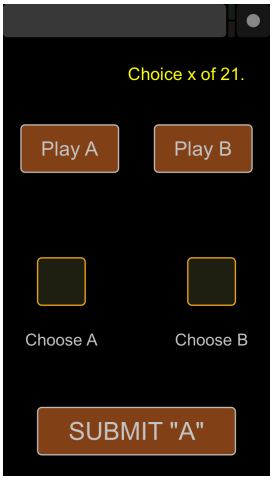


Figure 2. User interface for HRTF preference submission

3.2.1 *Trajectory*

The trajectory in figure 1 addresses considerations 1 and 2 specified in section 2.2. The horizontal plane orbit consisted of a single sustained note lasting approximately two seconds. Five short bursts in four static positions on the median plane used an even four-note phrase of around one second. The overall stimulus was a little under ten seconds. A key feature of the trajectory was that it passed through all eight virtual speaker locations, which takes advantage of VBAP's amplitude panning approach to spatialisation. Although the trajectory covered just 363 of 64,082 potential coordinates, this small minority focussed on the eight fundamental points from which all locations are rendered (consideration 1). The trajectory was also judged sufficiently short and simple enough to enable purely internalised A/B auditory comparison without reference to dynamic or interactive graphics (consideration 2).

3.2.2 *Selection process*

Participants used the GUI shown in figure 2 to compare trajectories and submit preferences. They were also given the diagram in figure 1 and an accompanying instruction:

> ***Which has the more convincing 3D effect, excerpt A or B?***
>
> *When comparing A and B, you may wish to consider:*
>
> - ***horizontal accuracy*** *during the orbit and at the four static central positions*
> - *sense of* ***spread*** *between front/back and up/down positions*
> - *sense of* ***distance*** *or how "outside of your head" the sounds seem*

Participants completed one example response to check their understanding of the task before starting. For each response, both the time elapsed and outcome of each comparison was logged automatically. Participants were allowed to listen to either trajectory as many times as they wished, but were forced to listen to A and B at least once in their entirety, before response buttons became enabled.

Figure 3. Localisation testing setup showing equipment worn, personalised calibration point and interaction mode

Both the sequence of comparisons and the order of A/B pairs were randomised for each participant, at all three sittings. For each of their three sessions, the HRTF sets chosen and rejected most often were designated as the preferred and least favoured options. In the event of a draw one of the tied sets was picked at random. These two designations were then used as the best and worst fitting HRTF sets in participants' subsequent localisation test.

3.3 Part 2: Interactive localisation test

The localisation test was conducted following a break of around five minutes. Before starting, the head-tracker was calibrated to a personalised position measured and agreed as approximately directly ahead and level with their eyeline and therefore considered as 0° azimuth, 0° elevation. Figure 3 shows the physical setup of the test environment. The localisation stimulus used 20 seconds of continuous music from the same recording used in part one [33].

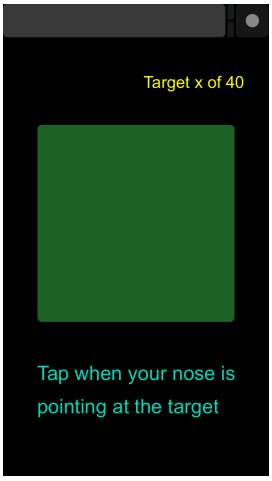


Figure 4. User interface for localisation test submission

3.3.1 Target locations

Localiation targets were divided into three strata, so that anticipated shortcomings in upper and lower hemisphere rendering cues could be evaluated independently:

- **at 45° elevation** – seven azimuths of -153°, -102°, -51°, 0°, 51°, 102° and 153°
- **at 0° elevation** – six of the azimuths stated above (0° was not used)
- **at -45° elevation** – the seven azimuths stated above

3.3.2 Localisation process

The test used egocentric head-pointing to report perceived source position, comparable to [34]. Participants used the simple GUI in figure 4 with the instruction:

> ***Where is the target sound source?***
>
> *Find the location of the target sound. Point your nose towards what you hear to be the source position.*
>
> *The source will be from somewhere around you and sometimes above or below your ear level. In some cases, you might need to rotate in your seat and/or tilt your head to point accurately.*

Participants completed two example responses to check that they understood what was required before starting the task. For each response, both the time elapsed and variance in head position from target location was logged automatically (as azimuth and elevation co-ordinates). Participants were allowed as much time as they needed to respond for each target. The 20 second excerpt continued on a loop, if necessary, until they registered a response, after which time the next target location began automatically.

Both the sequence of 20 co-ordinates and the order of the two groupings (preferred and least favoured HRTF set) were randomised for each participant, at all three sittings.

Systematic Disagreement	Weak Agreement	Fair or Good Agreement
C -0.306 (n)	*A* 0.349 (n)	*B* 0.429 (n)
D -1.080 (o)	*G* 0.186 (n)	*E* 0.743 (p)
H -0.095 (o)	*K* 0.075 (n)	*F* 0.437 (n)
J -0.418 (o)	*O* 0.342 (p)	*I* 0.715 (n)
L -0.840 (p)	*P* 0.380 (o)	*N* 0.726 (p)
M -0.795 (n)	*Q* 0.142 (o)	*R* 0.648 (o)
T -1.151 (p)	*U* 0.258 (o)	*S* 0.510 (o)

Table 1. HRTF selection reliability values and category for each participant (A-T), including binaural experience indicator (n = none; o = occasional; p = practised)

Therefore, a total of 120 data points was recorded for each participant using between a minimum of two (in the event of perfectly repeated best and worst selections) and maximum of six different HRTF sets.

4. RESULTS

All participants completed three study sessions at least 48 hours apart. During the session and on later examination of data, it became evident that one participant had not understood the requirements of the localisation task and had provided responses that did not actively seek out the position of the sound source. This participant's data is reflected in the analysis that follows in part one, but not in part two.

4.1 Part one: HRTF selection outcomes

Between the 21 participants, 63 HRTF selection procedures were completed. The average time taken across these was a mean of 13 and median of 11.8 minutes.

4.1.1 Ranking method

For each session, the outcomes of a participant's comparisons were translated into rank order based on the number of times each HRTF set was selected (a maximum of six and minimum of zero occasions). Tied HRTF sets were given a shared ranking at the highest jointly occupied position. So, for example, a ranking list of 1,2,3,4,4,4,7 reflects three HRTF sets gaining a score equal to fourth place.

4.1.2 Intra-class correlation measurement

Intra-class correlation (ICC) is a statistical approach used for measuring consistency between different raters to verify the robustness of a rating system [35]. ICC has been used previously to evaluate the reliability of repeated HRTF set ratings expressed by the same raters [26]. The HRTF selection reliability established for each participant via ICC is presented in table 1. Calculation of ICC was achieved using the R statistical computing package, according to the guidance and numerical outcome classifications provided in [35], where: less than 0.0 represents lower than chance levels of agreement (systematic disagreement); between 0.0 and less than 0.4 is an above chance (but weak) level of agreement; from 0.4 to less than 0.6 indicates fair agreement; between 0.6 and less than

0.75 shows good agreement; 0.75 and beyond constitutes excellent agreement.

Details provided by participants about their musical training, headphone listening habits and level of prior exposure to binaural audio were analysed by the groups in table 1 using chi-square and multiple linear regression tests. No relationship was evident between selection consistency and any factor. Each participant's level of experience with binaural audio is shown in table 1 for reference.

4.2 Part two: Interactive localisation outcomes

Three factors mean a reasonable degree of error was to be expected, particularly at the upper and lower strata (45° / -45° elevation). Firstly, even under optimal acoustic conditions, localisation blur of broadband sound immediately in front of a listener is established to be in the order of +/- 3.6° for azimuth and +/- 9° for elevation [36]. Secondly, inaccuracy in head pointing orientation was an further contributor to response error. Bahu et al [34] suggest that, for sources with substantial vertical displacement (57°), head pointed localisation can introduce mean unsigned error of 3° in azimuth and 12° in elevation. Thirdly, sparseness of the virtual speaker array in the upper and lower binaural hemisphere would have degraded spatial representation of sources originating in these areas far beyond optimal acoustic conditions [28, 30].

Given these constraints, minimum standards of accuracy were established to evaluate localisation outcomes. For azimuth, a tolerance of +/-15° was used to test whether the rendering system could provide reliable interactive presentation of sound sources at a minimum lateral separation of 30°. For elevation, a +/-22.5° threshold was applied to test simply whether users could reliably distinguish between sources located above, below and on the horizontal plane.

4.2.1 Influence of HRTF selection

Quality of HRTF fit could have impacted both response accuracy and time. Figure 5 shows the distribution of participant outcomes for best and worst HRTFs. Plots show the distribution of participants' overall azimuth and elevation success rate and their mean response duration, for each stratum (45°, 0° and -45°). If there were objective interaction benefits to the HRTF selection procedure, we would expect to see higher successful identification rates and lower mean response times. This is only evidenced clearly in relation to azimuth accuracy in the upper hemisphere (upper and middle plots of column one in figure 5). A Wilcoxon signed rank test confirmed significant improvement in accuracy of azimuth for sources placed on the horizontal plane, when using a preferred HRTF set compared to least preferred ($p = 0.047$). The same non-parametric test for significance did not uncover any other effects from using the best judged HRTF set, for any of the remaining eight metrics in figure 5.

Further analysis was conducted to evaluate the influence of HRTF selection consistency on localisation performance. A Kruskal-Wallis test was conducted between the participant groupings shown in table 1 (but without the participant identified in 4, who was within the 'Weak Agreement' group) and the same nine metrics reflected in figure 5. Significant difference was found in elevation accuracy at 0° ($p = 0.008$, $\chi^2 = 9.575$). Post-hoc Tukey-Kramer analysis identified that the 'Systematic Disagreement' group performed with significantly better accuracy against this metric than the 'Weak' group.

4.2.2 Influence of learning effects

Figure 6 represents the distribution of participant outcomes when viewed as the first and second halves of their sessions (irrespective of best/worst HRTF sequencing, which was always randomised). A Wilcoxon signed rank test confirmed significant improvement in accuracy of elevation identification with sources placed at 45°, for responses given in the second half of localisation trials ($p = 0.041$). The same non-parametric test further identified that responses at 45° ($p = 0.003$) and 0° ($p < 0.001$) were quicker in the second half, without any detrimental impact on azimuth or elevation precision.

5. DISCUSSION

We return to the four criteria put forward in section 2 for evaluating the success of an HRTF selection system.

5.1 Reliability

A third of participants demonstrated a fair to good level of consistency in the rankings that resulted between their three HRTF selection sessions. A further third showed some tendency towards repeating their patterns of selection beyond chance level. The final third returned sets of rankings that actively diverged from each other to a greater degree than chance level.

Although absolute values and proportions of participants between these groups do not indicate a mechanism that could yet be described as reliable, for a significant minority it was possible to attain outcomes that were repeatable to an acceptable level. Given the holistic nature of the comparison judgement (simultaneously considering azimuth, elevation and externalisation) and speed of the overall selection process, the approach shows substantial potential for further development towards more reliable usage. More detailed analysis of the selection process and localisation data presented here will help to identify how the stimulus, trajectory or written guidance outlined in section 3 could be further simplified to focus the comparison process.

5.2 Validity

Analysis showed apparent benefits to azimuth localisation accuracy in the upper hemisphere from preferred HRTF selection, which was significant along the horizontal plane. It is unsurprising that preferred HRTFs were of most benefit across this dimension, where five of the eight virtual speakers reside. Although this finding validates the selection approach in one respect, it is notable that positive elevation detection was increased by general exposure to the localisation task (albeit from a low starting base). This improvement and accompanying increases in response speed occurred independently of best or worst HRTF usage. The

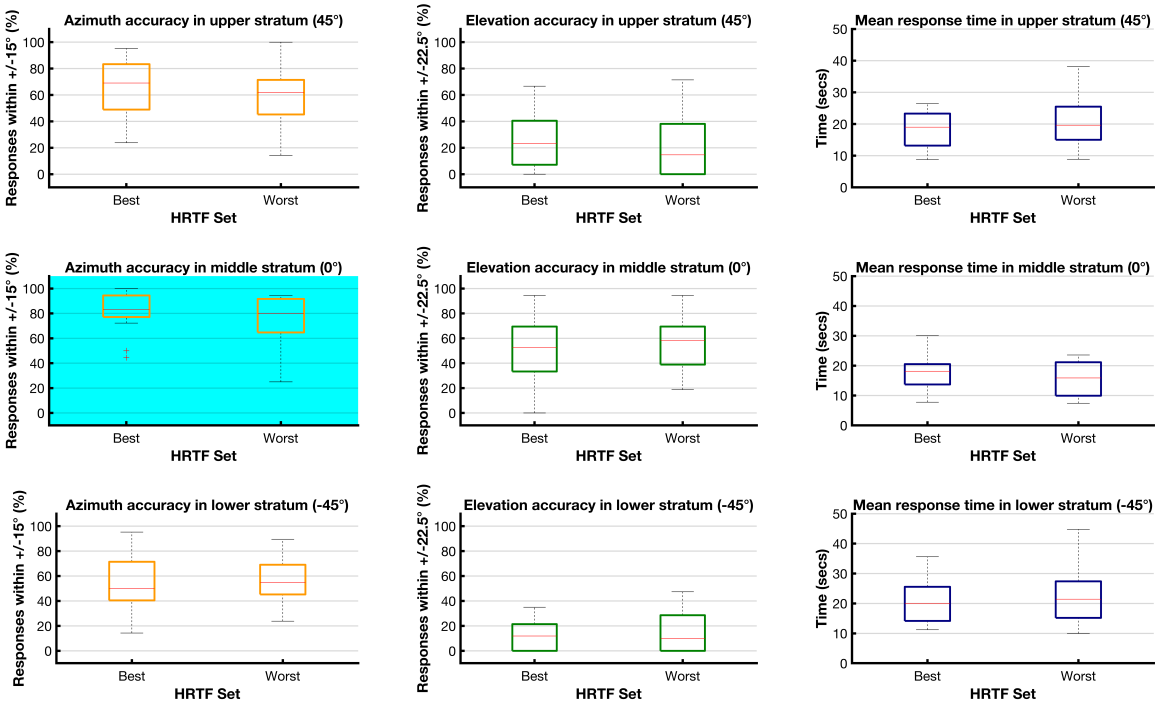


Figure 5. Distributions of participant azimuth/elevation success rates and mean response times, by HRTF preference*

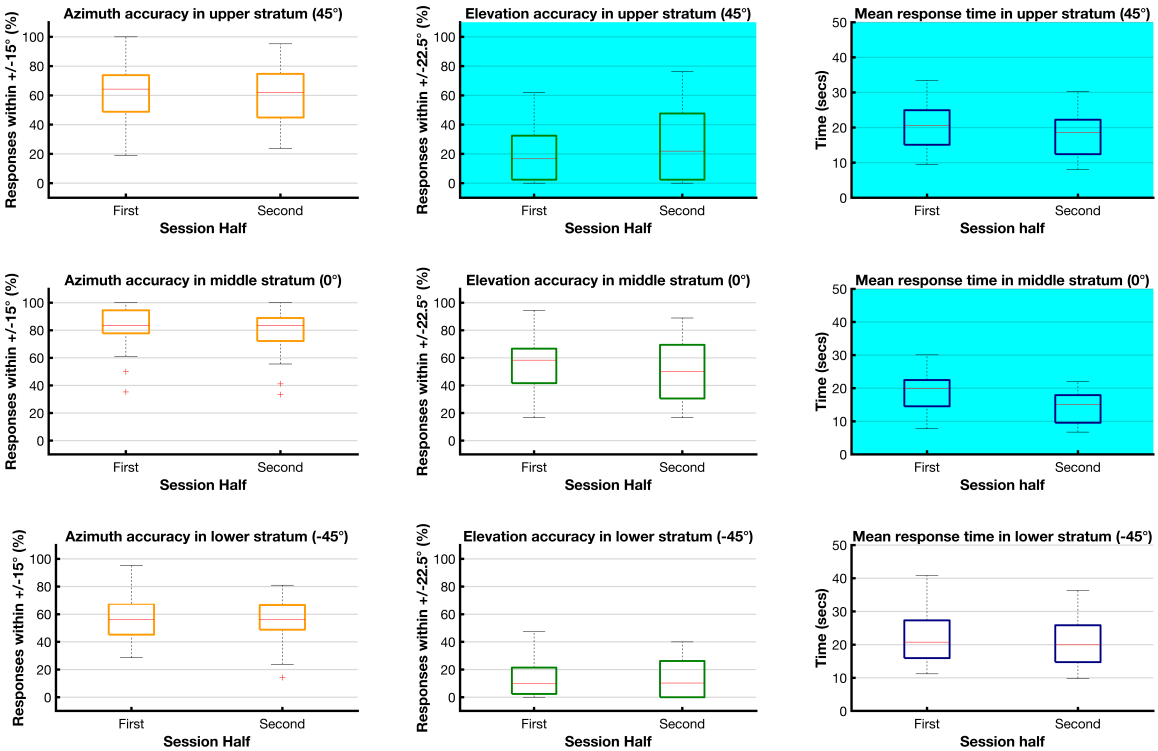


Figure 6. Distributions of participant azimuth/elevation success rates and mean response times, by sequence*

** Plots with blue background indicate significant difference between distributions*

selection routine might therefore be validly applied in tandem with a structured pre-exposure phase to optimise perceptual experience.

It should also be noted that no meaningful statistical relationship was found between participants' HRTF ranking consistency and localisation performance. Significantly improved elevation accuracy was found in the 'Systematic Disagreement' group for sources on the horizontal plane. However, this apparent strength is actually a by-product of that group returning a greater overall proportion of responses that neglected vertical localisation and remained overly focussed at 0° elevation. The group was less likely to have noticed vertically displaced sources and performed particularly poorly in elevation accuracy at heights of 45° and -45°.

5.3 Usability

It is notable that the two most reliable raters judged themselves to be practised in binaural listening. However, there was no significant advantage to ranking consistency found through statistical analysis of musical training, headphone listening habits or prior binaural exposure. Moreover, some of those with only occasional and even no binaural experience were able to achieve fair or good levels of repeatability.

5.4 Efficiency

An average completion time between 12 and 13 minutes can be regarded as within the realms of an acceptable duration for single-time calibration of a 3D end-user system.

6. CONCLUSION

A new generation of headsets is enabling various forms of auditory mixed reality with interactive spatial sound. The efficacy of those binaural experiences will be at least partly impacted by the degree to which rendered scenes fit the perceptual profile of individual users. To date, there is no established method for personalising the HRTFs deployed for end-users of audio virtual or augmented reality in mobile contexts. The approach outlined here begins to address this issue and its first iteration has shown promising outcomes against previously identified evaluation criteria.

Acknowledgments

This work was supported by EPSRC and AHRC under the grant EP/L01632X/1 (Centre for Doctoral Training in Media and Arts Technology).

7. REFERENCES

[1] D. R. Begault, *3D sound for virtual reality and multimedia*, 1st ed. London, UK: Academic Press Limited, 1994.

[2] B. B. Bederson, "Audio augmented reality: a prototype automated tour guide," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Denver, Colorado, USA, 5 1995, pp. 210–211.

[3] M. Cohen, S. Aoki, and N. Koizumi, "Augmented audio reality : telepresence / VR hybrid acoustic environments," in *IEEE International Workshop on Robot and Human Communication*, Tokyo, Japan, 11 1993, pp. 361–364.

[4] N. Sawhney and C. Schmandt, "Nomadic Radio: speech and audio interaction for contextual messaging in nomadic environments," *ACM Transactions on Computer-Human Interaction*, vol. 7, no. 3, pp. 353–383, 2000.

[5] Y. Vazquez-Alvarez, M. P. Aylett, S. A. Brewster, R. Von Jungenfeld, and A. Virolainen, "Designing interactions with multilevel auditory displays in mobile audio-augmented reality," *ACM Transactions on Computer-Human Interaction*, vol. 23, no. 1, pp. 1–30, 2016.

[6] M. McGill, S. Brewster, D. McGookin, and G. Wilson, "Acoustic transparency and the changing soundscape of auditory mixed reality," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Honolulu, Hawaii,USA, 4 2020, pp. 1–16.

[7] A. N. Nagele, V. Bauer, P. G. T. Healey, J. D. Reiss, H. Cooke, T. Cowlishaw, C. Baume, and C. Pike, "Interactive audio augmented reality in participatory performance," *Frontiers in Virtual Reality*, vol. 1, no. February, pp. 1–14, 2021.

[8] R. Shukla, "Voice-led interactive exploration of audio," BBC Research and Development, Tech. Rep. November, 2019.

[9] M. Geronazzo, A. Rosenkvist, D. S. Eriksen, C. K. Markmann-Hansen, J. Køhlert, M. Valimaa, M. B. Vittrup, and S. Serafin, "Creating an audio story with interactive binaural rendering in virtual reality," *Wireless Communications and Mobile Computing*, vol. 2019, pp. 1–14, 2019.

[10] A. Roginska, "Binaural audio through headphones," in *Immersive sound: The art and science of binaural and multi-channel audio*, 1st ed., A. Roginska and P. Geluso, Eds. New York, NY, USA: Routledge, 2018, ch. 4, pp. 88–123.

[11] M. Geronazzo, S. Spagnol, and F. Avanzini, "Estimation and modeling of pinna-related transfer functions," in *Proceedings of the 13th International Conference on Digital Audio Effects*, Graz, Austria, 9 2010, pp. 1–8.

[12] A. Meshram, R. Mehra, H. Yang, E. Dunn, J.-M. Franm, and D. Manocha, "P-HRTF: Efficient personalized HRTF computation for high-fidelity spatial sound," in *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Munich, Germany, 9 2014, pp. 53–61.

[13] S. Spagnol, "HRTF selection by anthropometric regression for improving horizontal localization accuracy," *IEEE Signal Processing Letters*, vol. 27, pp. 590–594, 2020.

[14] R. Pelzer, M. Dinakaran, F. Brinkmann, S. Lepa, P. Grosche, and S. Weinzierl, "Head-related transfer function recommendation based on perceptual similarities and anthropometric features," *The Journal of the Acoustical Society of America*, vol. 148, no. 6, pp. 3809–3817, 2020.

[15] B. U. Seeber and H. Fastl, "Subjective selection of non-individual head-related transfer functions," in *Proceedings of the 2003 International Conference on Auditory Display*, Boston, MA, USA, 7 2003, pp. 259–262.

[16] A. Roginska, G. H. Wakefield, and T. S. Santoro, "User selected HRTFs: reduced complexity and improved perception," in *Undersea Human System Integration Symposium*, Providence, RI, USA, 7 2010, pp. 1–14.

[17] R. Shukla, R. Stewart, A. Roginska, and M. Sandler, "User selection of optimal HRTF sets via holistic comparative evaluation," in *AES International Conference on Audio for Virtual and Augmented Reality*, Redmond, WA, USA, 8 2018, pp. 1–10.

[18] E. M. Wenzel, M. Arruda, D. J. Kistler, and F. L. Wightman, "Localization using nonindividualized head-related transfer functions," *The Journal of the Acoustical Society of America*, vol. 94, no. 1, pp. 111–123, 1993.

[19] H. Møller, M. F. Sørensen, C. B. Jensen, and D. Hammershøi, "Binaural technique: Do we need individual recordings?" *Journal of the Audio Engineering Society*, vol. 44, no. 6, pp. 451–469, 1996.

[20] D. R. Begault, E. M. Wenzel, and M. R. Anderson, "Direct comparison of the impact of head tracking, reverberation, and individualized head-related transfer functions on the spatial perception of a virtual speech source." *Journal of the Audio Engineering Society*, vol. 49, no. 10, pp. 904–916, 2001.

[21] A. Roginska, T. S. Santoro, and G. H. Wakefield, "Stimulus-dependent HRTF preference," in *129th Audio Engineering Society Convention*, San Francisco, CA, USA, 11 2010, pp. 1–11.

[22] Y. Wan, A. Zare, and K. Mcmullen, "Evaluating the consistency of subjectively selected head-related transfer functions (HRTFs) over time," in *AES 55th International Conference: Spatial Audio*, Helsinki, Finland, 8 2014, pp. 1–8.

[23] D. Schönstein and B. F. G. Katz, "Variability in perceptual evaluation of HRTFs," *Journal of the Audio Engineering Society*, vol. 60, no. 10, pp. 783–793, 2012.

[24] A. Andreopoulou and A. Roginska, "Evaluating HRTF similarity through subjective assessments: factors that can affect judgment," in *Joint 40th International Computer Music Conference & 11th Sound and Music Computing Conference*. Athènes, Greece: Michigan Publishing, 9 2014, pp. 1375–1381.

[25] A. Andreopoulou and B. F. G. Katz, "Investigation on subjective HRTF rating repeatability," in *140th Audio Engineering Society Convention*, Paris, France, 6 2016, pp. 1–10.

[26] C. Kim, V. Lim, and L. Picinali, "Investigation into consistency of subjective and objective perceptual selection of non-individual head-related transfer functions," *Journal of the Audio Engineering Society*, vol. 68, no. 11, pp. 819–831, 2020.

[27] V. Pulkki, "Virtual sound source positioning using vector base amplitude panning," *Journal of the Audio Engineering Society*, vol. 45, no. 6, pp. 456–466, 1997.

[28] R. Shukla, I. T. Radu, M. Sandler, and R. Stewart, "Real-time binaural rendering with virtual vector base amplitude panning," in *AES International Conference on Immersive and Interactive Audio*, York, UK, 3 2019, pp. 1–10.

[29] V. Pulkki, "Evaluating spatial sound with binaural auditory model," in *Proceedings of the 2001 International Computer Music Conference*. Havana, Cuba: Michigan Publishing, 9 2001, p. 73–76.

[30] R. Baumgartner and P. Majdak, "Modeling localization of amplitude-panned virtual sources in sagittal planes," *AES: Journal of the Audio Engineering Society*, vol. 63, no. 7-8, pp. 562–569, 2015.

[31] O. Warusfel, "Listen HRTF database," 2003. [Online]. Available: http://recherche.ircam.fr/equipes/salles/listen/index.html

[32] B. F. G. Katz and G. Parseihian, "Perceptually based head-related transfer function database optimization," *The Journal of the Acoustical Society of America*, vol. 131, no. 2, pp. EL99–EL105, 2012.

[33] V. Hansen and G. Munch, "Making recordings for simulation tests in the Archimedes project," *Journal of the Audio Engineering Society*, vol. 39, no. 10, pp. 768–774, 1991.

[34] H. Bahu, T. Carpentier, M. Noisternig, and O. Warusfel, "Comparison of different egocentric pointing methods for 3D sound localization experiments," *Acta Acustica united with Acustica*, vol. 102, no. 1, pp. 107–118, 2016.

[35] K. A. Hallgren, "Computing inter-rater reliability for observational data: an overview and tutorial," *Tutorials in Quantitative Methods for Psychology*, vol. 8, no. 1, pp. 23–34, 2012.

[36] J. Blauert, *Spatial hearing: the psychophysics of human sound localization*, revised ed. Cambridge, Massachusetts: MIT Press, 1997.

FINITE-DIFFERENCE SIMULATION OF LINEAR PLATE VIBRATION WITH DYNAMIC DISTRIBUTED CONTACT

Maarten VAN WALSTIJN[1,2], **Abhiram BHANUPRAKASH**[1,2], and **Paul STAPLETON**[1,3]

[1] *Sonic Arts Research Centre*, **Queen's University Belfast**, UK
[2] *School of Electronics, Electrical Engineering and Computer Science*, **Queen's University Belfast**, UK
[3] *School of Arts, English, and Languages*, **Queen's University Belfast**, UK

ABSTRACT

The potential of physics-based synthesis algorithms for developing computer-based musical instruments relies on the inclusion of articulatory elements that enable physically plausible and musically meaningful interactions. In this paper, non-excitational interaction with a rectangular vibrating plate is modelled through time variation in distributed contact parameters. For numerical simulation a finite difference approach is taken, enabling efficient modelling of local interactions. Comparison between the continuous- and discrete-domain system power balances confirms conditional stability and a match in the source terms due to parameter time-variance. The methodology is exemplified with a few case studies, and its potential for application in the design of a virtual-acoustic plate-based musical instrument is discussed.

1. INTRODUCTION

Music performed with mechano-acoustic instruments is often articulated through physical contact, either directly between the musician and a resonating element or via an extension (e.g. a mallet, bow, or slide). Such embodied interaction generally serves two kinds of purposes. The first is to inject energy into the instrument's vibrating parts, such as plucking a string or striking a membrane. The second is to alter the vibrating properties of the instrument, for example holding down a string against a fretboard to achieve a specific pitch. Some forms of contact serve both purposes simultaneously, e.g. guitar tapping [1].

In the case of 2-D vibrating systems, such as membranes and plates featuring in percussion instruments, contact for the second purpose is often of a distributed nature, and generally involves *damping*, *clamping*, and *mass-loading*. For any specific form of contact, typically one of those phenomena dominates. For example, Figure 2(a) displays a common way to damp the sound of a vibrating cymbal, in which the hand's strong damping is accompanied by small additional stiffness (which can be considered as clamping) and mass-loading. The spectrogram in Figure 2(d) of a recorded cymbal strike with hand damping applied at around $t = 1.2$s shows how the partials may not be damped to the same extent, and also shows slight lowering of some of the resonance frequencies. Figure 2(b) shows a partially clamped vibraphone bar, introducing spring-like distributed restoring forces over the clamped region, and altering the resonant characteristics. As can be seen in the spectra shown in Figure 2(e), the specific resonance alterations depend on where the system is clamped. Somewhat similar to placing a capo on the guitar neck, this form of contact is conducted more on the level of design and reconfiguration than performatory action. Figure 2(c) shows the partial submersion of a small round gong in water, effectively introducing strong mass-loading on the submerged region, which lowers the resonance frequencies. Modulating the plate's resonance frequencies by dynamically changing the amount of submerged plate surface is an established performance practice with gongs and tam-tams (see, e.g. [2]). Figure 2(f) shows the spectrogram of a gong strike followed by gradually lifting the gong upwards and then back downwards, thus effecting a temporary decrease in submerged surface area.

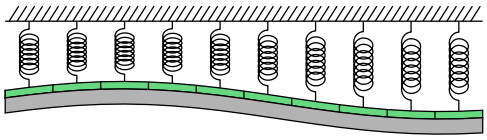


Figure 1. Schematic representation of the locally reacting contact layer. The grey object represents the plate, and the green elements represent additional surface density. The springs represent additional stiffness and damping per unit area.

From a music articulation perspective, dynamic contact is of particular interest. The three aforementioned phenomena are therefore considered here as time-varying.

This paper focuses on the simulation of such time-varying, non-excitational distributed contact, with a view to facilitate physically plausible contact with intuitive parameter control in physics-based sound synthesis, using a rectangular plate model with free edges as a testbed 2-D resonator. The approach is to model the contact in terms of additional damping, stiffness, and mass-loading terms (see Figure 1), in effect adding what is known in room acoustics as a locally-reacting layer. Similar to earlier work on modelling string excitation [3, 4], dynamic contact is emulated via time variation of the local contact parameters. For example, initiating contact of a hand with

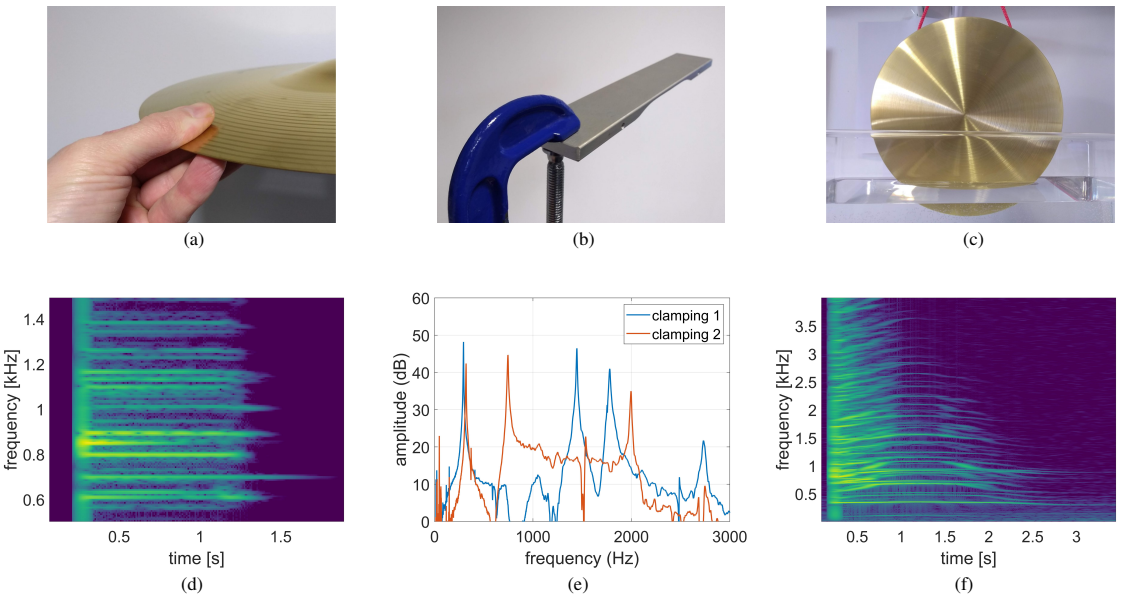


Figure 2. Examples of distributed contact in 2-D musical resonators: (a) regional damping of a cymbal via local hand contact, (b) regional clamping of a vibraphone bar, (c) regional mass loading of a round gong by submerging it partially in water, (d) spectrogram of a cymbal strike with hand damping at $t = 1.2$ s, (e) amplitude spectra of a struck vibraphone bar clamped at two different positions, (f) spectrogram of a gong with time-varying water submersion.

the plate is modelled by ramping up from zero the spring stiffness and damping per unit area as well as the surface density. For static values of the contact parameters the model is grounded in Newtonian physics, but the specification of how the parameters vary to emulate dynamic contact is, due to the simplifications involved, more phenomenological. It would be more physically correct to model the motion of the hand or other external object separately, allowing also to simulate collisions, such as recently reported in the context of slide-string articulation in [5]. The reasons for not taking a more complex approach are (a) collisions have a limited role in most practical examples of non-excitational contact (except for sympathetic vibrations, which can be modelled separately), (b) it would require updating a large set of additional mass positions, adding significant computational burden, and (c) in a real-time scenario, the distance between the object and the 2-D resonator would have to be sensed across the spatial coordinates, which is a challenge in its own right. For the simpler model chosen, only a pressure map needs to be sensed, which can be done with available sensing devices, such as the Sensel Morph [1] ; the local contact parameters may then be specified as an implicit function of the local control pressure.

Regarding the modelling of plate vibrations, i.e. the system without contact terms, the equations of motion given in Section 2 and their discretisation and associated energy analysis presented in Section 3 are equivalent to those presented in earlier work on plate modelling [6, 7]. The novel aspect in these sections revolves around the introduction of the time-varying contact terms and their discretisation.

[1] https://sensel.com/

2. PLATE MODEL WITH CONTACT

Let $\partial_x u$ and $d_x u$ denote the partial and total derivative of u with respect to x, respectively.

2.1 System Equations

Consider transversal vibrations of a thin plate with damping and distributed contact terms:

$$\rho h \partial_t^2 u = -D\Delta^2 u + 2\rho h \left[\sigma_2 \Delta - \sigma_0\right] \partial_t u + \psi F_\mathrm{e} \underbrace{-\partial_t(\bar{\lambda}\partial_t u) - \bar{k}u - 2\rho h \bar{\sigma} \partial_t u}_{\text{contact}}, \quad (1)$$

where $u = u(x, y, t)$ denotes transversal displacement, ρ is mass density, and σ_0 and σ_2 are damping constants. The parameter $D = \frac{1}{12} E h^3/(1-\nu^2)$ can be considered as characterising the plate stiffness, where E is the Young's modulus, h is the plate thickness, and ν is the Poisson's ratio. The first term on the right-hand side of (1) can be written

$$\begin{aligned} -D\Delta^2 u &= -D\left(\partial_x^4 + \partial_y^4 + 2\partial_x^2\partial_y^2\right) u, \\ &= \partial_x^2 m_x + \partial_y^2 m_y + 2\partial_x\partial_y m_{xy}, \end{aligned} \quad (2)$$

where the plate bending moments

$$m_x = -D\left(\partial_x^2 u + \nu \partial_y^2 u\right), \quad (3)$$

$$m_y = -D\left(\partial_y^2 u + \nu \partial_x^2 u\right), \quad (4)$$

$$m_{xy} = -D\left(1-\nu\right)\partial_x\partial_y u, \quad (5)$$

are introduced due to their usefulness in defining boundary conditions and in performing energy analysis. The term $\psi = \psi(x, y, t) = \delta_D(x - x_\mathrm{e}(t), y - y_\mathrm{e}(t))$ defines the position of the driving force $F_\mathrm{e} = F_\mathrm{e}(t)$.

The lower line in (1) contains three time-varying terms due to distributed contact, in accordance with Figure 1. The contact surface density $\bar{\lambda} = \bar{\lambda}(x, y, t)$, stiffness $\bar{k} = \bar{k}(x, y, t)$ and damping $\bar{\sigma} = \bar{\sigma}(x, y, t)$ are each modelled here as being dependent on a non-dimensional distributed control pressure map $\bar{p} = \bar{p}(x, y, t)$:

$$\bar{\lambda} = a_\lambda \bar{p}, \quad \bar{k} = a_k \bar{p}, \quad \bar{\sigma} = a_\sigma \bar{p}(1 + \bar{g}) + \sigma_0 \bar{g}, \quad (6)$$

where $\bar{g} = \bar{\lambda}/(\rho h)$ and $a_i \geq 0$, $i \in \{\lambda, k, \sigma\}$ can be tuned to suit specific interaction design purposes, and with $0 \leq \bar{p} \leq 1$. The control pressure is a map that - for real-time application - could be sensed with a force-sensitive electronic pad, such as the Sensel Morph. Grouping common terms, one can re-write (1) as

$$\partial_t \left[\left(\rho h + \bar{\lambda} \right) \partial_t u \right] = - \left[D\Delta^2 + \bar{k} \right] u + \psi F_\mathrm{e} + 2\rho h \left[\sigma_2 \Delta - (\bar{\sigma} + \sigma_0) \right] \partial_t u. \quad (7)$$

Hence the effective frequency-independent damping is

$$\sigma_{0,\mathrm{eff}} = \frac{2\rho h \left(\bar{\sigma} + \sigma_0 \right)}{2 \left(\rho h + \bar{\lambda} \right)} \overset{(6)}{=} \sigma_0 + a_\sigma \bar{p}, \quad (8)$$

which shows how the specific form for $\bar{\sigma}$ in (6) facilitates direct control of the effective additional local damping. Eq. (7) also brings to light how practical ranges for each of the contact parameters are relative to the 'corresponding' plate parameters.

2.2 Boundary Conditions

Free boundary conditions are imposed, mainly to enable a natural way of damping by grabbing the edge of the plate (like in cymbals, see Figure 2a). For a rectangular plate of size $L_x \times L_y$, at the edges we have [8]

$$x = 0, L_x: \quad m_x = 0, \quad \partial_x m_x + 2\partial_y m_{xy} = 0, \quad (9)$$
$$y = 0, L_y: \quad m_y = 0, \quad \partial_y m_y + 2\partial_x m_{xy} = 0, \quad (10)$$

Through substitution this can be written as

$$x{=}0, L_x: \quad \partial_x^2 u + \nu \partial_y^2 u = 0, \quad \partial_x^3 u + (2-\nu)\partial_x \partial_y^2 u = 0, \quad (11)$$
$$y{=}0, L_y: \quad \partial_y^2 u + \nu \partial_x^2 u = 0, \quad \partial_y^3 u + (2-\nu)\partial_y \partial_x^2 u = 0. \quad (12)$$

An additional condition is required at corners:

$$m_{xy} = 0, \implies \partial_x \partial_y u = 0. \quad (13)$$

2.3 Energy Analysis

Given the functions $f(x, y, t)$ and $g(x, y, t)$, let's define the plate domain

$$\mathcal{P} = \{(x, y) \in \mathbb{R} \,|\, 0 \leq x \leq L_x, 0 \leq y \leq L_y\} \quad (14)$$

and the associated inner product and norm

$$\langle f, g \rangle_\mathcal{P} = \int_{y=0}^{L_y} \int_{x=0}^{L_x} f(x, y, t)\, g(x, y, t)\, dx dy, \quad (15)$$
$$\|f\|_\mathcal{P} = \sqrt{\langle f, f \rangle_\mathcal{P}}, \quad (16)$$

which will allow compact notation of energy terms. Multiplying (1) with $\partial_t u$ and integrating over the plate domain $\mathcal{P}$, one obtains the power balance

$$d_t \left(H_\mathrm{p} + H_\mathrm{c} \right) = P_\mathrm{e} + P_\mathrm{c} - Q_\mathrm{p} - Q_\mathrm{c}, \quad (17)$$

with the (non-negative) energy components

$$H_\mathrm{p} = \frac{1}{D(1-\nu^2)} \left[\tfrac{1}{2} \|m_x\|_\mathcal{P}^2 + \tfrac{1}{2} \|m_y\|_\mathcal{P}^2 - \nu \langle m_x, m_y \rangle_\mathcal{P} \right] + \tfrac{1}{2} \rho h \|\partial_t u\|_\mathcal{P}^2 + \frac{1}{D(1-\nu)} \|m_{xy}\|_\mathcal{P}^2, \quad (18)$$
$$H_\mathrm{c} = \tfrac{1}{2} \langle \bar{k} u, u \rangle_\mathcal{P} + \tfrac{1}{2} \langle \bar{\lambda}\, \partial_t u, \partial_t u \rangle_\mathcal{P}. \quad (19)$$

The power input terms (of indeterminate sign) are

$$P_\mathrm{e} = F_\mathrm{e} \langle \psi, \partial_t u \rangle_\mathcal{P}, \quad (20)$$
$$P_\mathrm{c} = \tfrac{1}{2} \langle u\, \partial_t \bar{k}, u \rangle_\mathcal{P} - \tfrac{1}{2} \langle \partial_t u\, \partial_t \bar{\lambda}, \partial_t u \rangle_\mathcal{P}. \quad (21)$$

and the (non-negative) damping terms are

$$Q_\mathrm{p} = 2\rho h \left(\sigma_0 \|\partial_t u\|_\mathcal{P}^2 + \sigma_2 \|\partial_t \nabla u\|_\mathcal{P}^2 \right), \quad (22)$$
$$Q_\mathrm{c} = 2\rho h \langle \bar{\sigma}\, \partial_t u, \partial_t u \rangle_\mathcal{P}. \quad (23)$$

In the above, the subscripts 'p', 'c', and 'e' indicate 'plate', 'contact', and 'excitation', respectively. As explained in [7], additional boundary terms in (17) obtained initially after integration by parts vanish for the boundary conditions in (9), (10), and (13).

3. NUMERICAL FORMULATION

3.1 Discretisation Operators

Employing the usual spatio-temporal gridding notation $u_{l,m}^n$ to denote u at time $t = n\Delta_t$ and position $x = l\Delta_x$, $y = m\Delta_x$, the following shift operators are defined:

$$\epsilon_{t+} u_{l,m}^n = u_{l,m}^{n+\frac{1}{2}}, \qquad \epsilon_{t-} u_{l,m}^n = u_{l,m}^{n-\frac{1}{2}}. \quad (24)$$

Elemental temporal difference and averaging operators can then be constructed as

$$\delta_t = \frac{\epsilon_{t+} - \epsilon_{t-}}{\Delta_t}, \qquad \mu_t = \frac{\epsilon_{t+} + \epsilon_{t-}}{2}, \quad (25)$$
$$\delta_{t+} = \frac{\epsilon_{t+}^2 - 1}{\Delta_t}, \qquad \mu_{t+} = \frac{\epsilon_{t+}^2 + 1}{2}, \quad (26)$$
$$\delta_{t-} = \frac{1 - \epsilon_{t-}^2}{\Delta_t}, \qquad \mu_{t-} = \frac{1 + \epsilon_{t-}^2}{2}. \quad (27)$$

Various finite-difference approximations can be achieved by directly applying or combining these elemental operators, e.g.

$$\delta_t^2 u_{l,m}^n = \frac{u_{l,m}^{n+1} - 2u_{l,m}^n + u_{l,m}^{n-1}}{\Delta_t^2} \approx (\partial_t^2 u)^*, \quad (28)$$
$$\delta_{t\cdot} u_{l,m}^n \coloneqq \delta_t \mu_t u_{l,m}^n = \frac{u_{l,m}^{n+1} - u_{l,m}^{n-1}}{2\Delta_t} \approx (\partial_t u)^*, \quad (29)$$
$$\mu_t^2 u_{l,m}^n = \frac{u_{l,m}^{n+1} + 2u_{l,m}^n + u_{l,m}^{n-1}}{4} \approx u^*, \quad (30)$$

where the asterisk denotes that the relevant quantity is evaluated at time $n\Delta_t$ and position $(l\Delta_x, m\Delta_x)$. Spatial operators can be defined analogously, by replacing t with x or y in the above, and using $\Delta_y = \Delta_x$ for convenient analysis and implementation [6]. A discrete Laplacian and biharmonic operator can be constructed as

$$\tilde{\Delta} = \delta_x^2 + \delta_y^2, \tag{31}$$

$$\tilde{\Delta}^2 = \delta_x^4 + 2\delta_x^2\delta_y^2 + \delta_y^4. \tag{32}$$

3.2 Product Identities

The following product identities hold for any two time series $f = f^n$ and $g = g^n$:

$$\mu_t\left(\mu_t f \cdot \mu_t g\right) \cdot \delta_{t\cdot} g = \tfrac{1}{2}\delta_t\left[\mu_t f \cdot (\mu_t g)^2\right] - \delta_{t\cdot} f \cdot \mu_{t+} g \cdot \mu_{t-} g, \tag{33}$$

$$\delta_t\left(\mu_t f\, \delta_t g\right) \cdot \delta_{t\cdot} g = \tfrac{1}{2}\delta_t\left[\mu_t f \cdot (\delta_t g)^2\right] + \delta_{t\cdot} f \cdot \delta_{t+} g \cdot \delta_{t-} g, \tag{34}$$

which is easily proven by direct evaluation of the terms to which discretisation operators are applied. These identities are useful in energy analysis regarding the terms with time-varying parameters. For the time-invariant terms, here relating to plate vibrations, use can be made of a handful of identities involving inner products (see, e.g. [6, 7]).

3.3 Numerical Scheme

The fully explicit numerical scheme employed here results from applying discretisation operators to (7) as follows:

$$\begin{aligned}\delta_t\left[\left(\rho h + \mu_t\bar{\lambda}^n_{l,m}\right)\delta_t u^n_{l,m}\right] = &-D\tilde{\Delta}^2 u^n_{l,m} + \psi^n_{l,m}F^n_{\mathrm{e}} \\ &- \mu_t\left(\mu_t\bar{k}^n_{l,m}\,\mu_t u^n_{l,m}\right) \\ &- 2\rho h\left[\bar{\sigma}^n_{l,m} + \sigma_0\right]\delta_{t\cdot}u^n_{l,m} \\ &+ 2\rho h\sigma_2\tilde{\Delta}\delta_{t-}u^n_{l,m},\end{aligned} \tag{35}$$

where $\psi^n_{l,m}$ is a discrete-domain version of the distribution function $\psi(x, y, t)$ obtained through bilinear deinterpolation. Dropping the indexes in the notation, e.g. $u = u^n_{l,m}$, one may write the first term on the right-hand side in terms of the discretised moments:

$$-D\tilde{\Delta}^2 u = \delta_x^2 m_x + \delta_y^2 m_y + 2\delta_{x^-}\delta_{y^-} m_{xy}, \tag{36}$$

where

$$m_x = -D\left(\delta_x^2 u + \nu\delta_y^2 u\right), \tag{37}$$

$$m_y = -D\left(\delta_y^2 u + \nu\delta_x^2 u\right), \tag{38}$$

$$m_{xy} = -D\left(1-\nu\right)\delta_{x+}\delta_{y+}u, \tag{39}$$

with $m_{xy} = m^n_{xy,l+\frac{1}{2},m+\frac{1}{2}}$. By substitution, we can recover the biharmonic operator in the form of (32).

3.4 Boundary Conditions

Using a non-centered approach for the boundaries, at the left and bottom edge one may specify [7]:

$$x{=}0: \quad m_x{=}0, \quad \delta_{x^-}m_x + 2e_{x^-}\delta_{y^-}m_{xy}{=}0, \tag{40}$$

$$y{=}0: \quad m_y{=}0, \quad \delta_{y^-}m_y + 2e_{y^-}\delta_{x^-}m_{xy}{=}0. \tag{41}$$

Through substitution we obtain:

$$x{=}0: \quad \delta_x^2 u + \nu\delta_y^2 u{=}0, \quad \delta_{x^-}\left[\delta_x^2 u + (2{-}\nu)\delta_y^2\right]u{=}0, \tag{42}$$

$$y{=}0: \quad \delta_y^2 u + \nu\delta_x^2 u{=}0, \quad \delta_{y^-}\left[\delta_y^2 u + (2{-}\nu)\delta_x^2\right]u{=}0, \tag{43}$$

which is analogous to (11,12). For the corner at $x = 0, y = 0$, a suitable numerical condition is

$$e_{x^-}e_{y^-}m_{xy} = 0, \implies \delta_{x^-}\delta_{y^-}u^n_{l,m} = 0. \tag{44}$$

Appropriately symmetric versions of the above conditions are applied to the other edges and corners.

3.5 Energy Analysis

For non-centered boundaries, the numerical inner product takes the form [6, 7]

$$\langle f, g\rangle_{\mathcal{P}} = \sum_{l=0}^{M_x}\sum_{m=0}^{M_y} f^n_{l,m}\, g^n_{l,m}\,\Delta_x^2. \tag{45}$$

where M_x and M_y define the grid size. After multiplying (35) with $\delta_{t\cdot}u$ and performing summation by parts, one obtains the power balance

$$\delta_t\left(H^n_{\mathrm{p}} + H^n_{\mathrm{c}}\right) = P^n_{\mathrm{e}} + P^n_{\mathrm{c}} - Q^n_{\mathrm{p}} - Q^n_{\mathrm{c}}, \tag{46}$$

where, making use of the product identities presented in Section 3.2, the energy components and power input terms can be specified as

$$\begin{aligned}H^{n+\frac{1}{2}}_{\mathrm{p}} = &\tfrac{1}{2}\rho h\|\delta_{t+}u\|_{\mathcal{P}} + \frac{1}{D(1-\nu)}\langle m_{xy}, e_{t+}m_{xy}\rangle_{\mathcal{P}} \\ &- \tfrac{1}{2}\rho h\sigma_2\Delta_t\left(\|\delta_{t+}\delta_{x+}u\|^2_{\mathcal{P}} + \|\delta_{t+}\delta_{y+}u\|^2_{\mathcal{P}}\right) \\ &+ \frac{1}{D(1-\nu^2)}\Big[\tfrac{1}{2}\langle m_x, e_{t+}m_x\rangle_{\mathcal{P}} \\ &+ \tfrac{1}{2}\langle m_y, e_{t+}m_y\rangle_{\mathcal{P}} - \tfrac{1}{2}\nu\langle m_x, e_{t+}m_y\rangle_{\mathcal{P}} \\ &- \tfrac{1}{2}\nu\langle e_{t+}m_x, m_y\rangle_{\mathcal{P}}\Big],\end{aligned} \tag{47}$$

$$\begin{aligned}H^{n+\frac{1}{2}}_{\mathrm{c}} = &\tfrac{1}{2}\langle\mu_{t+}\bar{k}\,\mu_{t+}u, \mu_{t+}u\rangle_{\mathcal{P}} \\ &+ \tfrac{1}{2}\langle\delta_{t+}u\,\mu_{t+}\bar{\lambda}, \delta_{t+}u\rangle_{\mathcal{P}},\end{aligned} \tag{48}$$

$$P^n_{\mathrm{e}} = F^n_{\mathrm{e}}\langle\psi, \delta_{t\cdot}u\rangle_{\mathcal{P}}, \tag{49}$$

$$P^n_{\mathrm{c}} = \tfrac{1}{2}\langle\mu_{t+}u\,\mu_{t-}u, \delta_{t\cdot}\bar{k}\rangle_{\mathcal{P}} - \tfrac{1}{2}\langle\delta_{t+}u\,\delta_{t-}u, \delta_{t\cdot}\bar{\lambda}\rangle_{\mathcal{P}}. \tag{50}$$

The damping terms are

$$\begin{aligned}Q^n_{\mathrm{p}} = &2\rho h\Big[\sigma_0\|\delta_{t\cdot}u\|^2_{\mathcal{P}} \\ &+ \sigma_2\left(\|\delta_{t\cdot}\delta_{x-}u\|^2_{\mathcal{P}} + \|\delta_{t\cdot}\delta_{y-}u\|^2_{\mathcal{P}}\right)\Big],\end{aligned} \tag{51}$$

$$Q^n_{\mathrm{c}} = 2\rho h\langle\bar{\sigma}\,\delta_{t\cdot}u, \delta_{t\cdot}u\rangle_{\mathcal{P}}. \tag{52}$$

As before with the continuous-domain power balance, additional boundary terms in (46) obtained initially after summation by parts vanish for the boundary conditions in (40), (41), and (44). The scheme can thus be said to be stable under the condition that the numerical system energy $H^{n+\frac{1}{2}} = H^{n+\frac{1}{2}}_{\mathrm{p}} + H^{n+\frac{1}{2}}_{\mathrm{c}}$ remains non-negative. Given that

$H_{\mathrm{c}}^{n+\frac{1}{2}} \geq 0$ without further conditions, this requires non-negativity of $H_{\mathrm{p}}^{n+\frac{1}{2}}$, which can be shown to hold if

$$\Delta_x \geq \Delta_x^{\min} = \sqrt{4\Delta_t \left(\sigma_2 + \sqrt{\sigma_2^2 + D/(\rho h)}\right)}. \quad (53)$$

In implementatons, this condition is met by setting $\Delta_x = \Delta_x^{\min}$ and adjusting the plate dimensions to fit to the grid:

$$L_x = \underbrace{\lfloor L_x'/\Delta_x \rceil}_{M_x} \Delta_x, \quad L_y = \underbrace{\lfloor L_y'/\Delta_x \rceil}_{M_y} \Delta_x, \quad (54)$$

where L_x' and L_y' are the initial target dimensions. Comparison of the above equations with those of section 2.3 reveals that, provided that the stability condition is satisfied, the numerical system mirrors the energy behaviour of its underlying continuous-domain counterpart, in that (a) the system is strictly dissipative in the absence of external forces and time-variation in the contact parameters, and (b) variation in $\bar{\lambda}$ or $\bar{k}$ results in articulatory power sources of matching form, i.e. (50) takes the form of a direct discretisation of (21).

3.6 Update Equation

For the scheme in (35) one can derive the general update equation:

$$\begin{aligned}\Big[c_0 + \bar{\zeta}_{l,m}^n + \bar{\kappa}_{l,m}^{n+\frac{1}{2}} + \bar{\gamma}_{l,m}^{n+\frac{1}{2}}\Big] u_{l,m}^{n+1} = \\ \Big[c_3 - (\bar{\kappa}_{l,m}^{n+\frac{1}{2}} + \bar{\kappa}_{l,m}^{n-\frac{1}{2}}) + (\bar{\gamma}_{l,m}^{n+\frac{1}{2}} + \bar{\gamma}_{l,m}^{n-\frac{1}{2}})\Big] u_{l,m}^n \\ + \Big[c_4 + \bar{\zeta}_{l,m}^n - \bar{\kappa}_{l,m}^{n-\frac{1}{2}} - \bar{\gamma}_{l,m}^{n-\frac{1}{2}}\Big] u_{l,m}^{n-1} \\ + c_2 v_{l,m}^n + c_5 v_{l,m}^{n-1} + c_1\Big[2s_{l,m}^n + w_{l,m}^n\Big] \\ + c_6\Big[\psi_{l,m}^n F_{\mathrm{e}}^n\Big]. \end{aligned} \quad (55)$$

where

$$v_{l,m}^n = u_{l+1,m}^n + u_{l-1,m}^n + u_{l,m+1}^n + u_{l,m-1}^n, \quad (56)$$

$$w_{l,m}^n = u_{l+2,m}^n + u_{l-2,m}^n + u_{l,m+2}^n + u_{l,m-2}^n, \quad (57)$$

$$s_{l,m}^n = u_{l+1,m+1}^n + u_{l-1,m+1}^n + u_{l+1,m-1}^n + u_{l-1,m-1}^n. \quad (58)$$

Note that for nodes on and near the boundary, ghost-nodes (nodes falling outside the domain) come into play in the expressions in (56), (57) and (58), and these need to be substituted for using the boundary conditions in (42), (43), and (44). The various coefficients in (55) are defined as

$$\bar{\kappa}_{l,m}^{n+\frac{1}{2}} = \tfrac{1}{4}\xi\mu_t \bar{k}_{l,m}^{n+\frac{1}{2}}, \qquad \bar{\gamma}_{l,m}^{n+\frac{1}{2}} = \mu_t \bar{g}_{l,m}^{n+\frac{1}{2}}, \quad (59)$$

$$c_0 = 1 + \zeta_0, \qquad c_1 = -K^2, \quad (60)$$

$$c_2 = 8K^2 + \zeta_2, \qquad c_3 = 2 - 20K^2 - 4\zeta_2, \quad (61)$$

$$c_4 = -1 + 4\zeta_2 + \zeta_0, \quad c_5 = -\zeta_2, \qquad c_6 = \xi, \quad (62)$$

with

$$K = \sqrt{\frac{D}{\rho h}}\frac{\Delta_t}{\Delta_x^2}, \quad \zeta_0 = \sigma_0 \Delta t, \qquad \zeta_2 = \frac{2\sigma_2 \Delta t}{\Delta_x^2},$$

$$\bar{\zeta}_{l,m}^n = \bar{\sigma}_{l,m}^n \Delta_t, \qquad \xi = \frac{\Delta_t^2}{\rho h}, \qquad \bar{g}_{l,m}^n = \frac{\bar{\lambda}_{l,m}^n}{\rho h}. \quad (63)$$

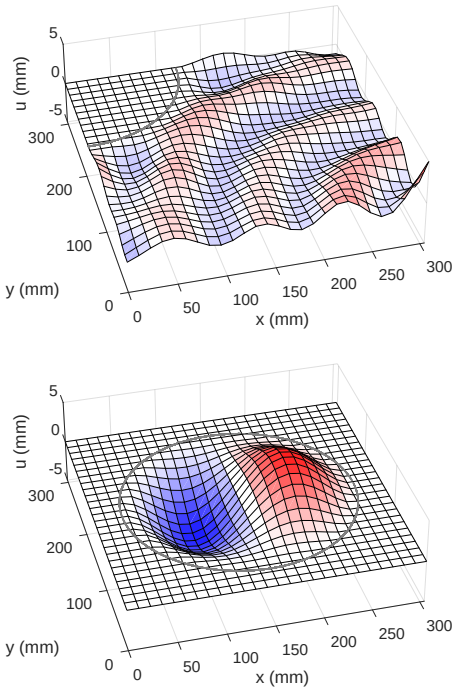


Figure 3. Examples of regional clamping of a square-shaped steel plate. Top: corner region clamp. Bottom: circular plate with clamped edges.

4. NUMERICAL EXPERIMENTS

In this section, the functionality of the model is demonstrated with a few examples. In all cases, the material parameters used are $E = 200$ GPa, $\rho = 8000$ kg/m^3 and $\nu = 0.3$, which are typical of steel. The simulations are run using $\Delta_t = 1/44100$ s. To help manage potential drift, a very small additional amount of clamping ($\bar{k} = 100$ N/m^3) is systematically applied across the plate.

4.1 Static Contact

4.1.1 Regional Clamping

Figure 3 shows snapshots of two simulations in which a specific region of a square plate was clamped by setting $\bar{k} = 10^{13}$ N/m^3 in that region. In the first experiment (see top plot in Figure 3), a small quarter-circular region at one of the corners was clamped, and the system was brought into vibration by driving the opposite corner with a sinusoid. For such a high $\bar{k}$ value, practically no oscillation can occur in the clamped region. Setting lower values facilitates softly clamped regions, which lead to different system resonances. In the second experiment (see bottom plot in Figure 3), the clamped region was chosen as lying outside a centrally-positioned circle, allowing the simulation of a clamped circular plate, albeit accepting approximations in terms of the domain shape due to the spatial discretisation. The system was driven off-center with the second lowest resonance frequency of the system, as such revealing the shape of its (1,1) normal mode.

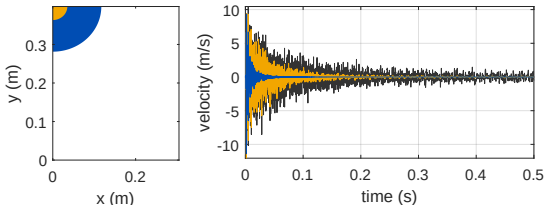


Figure 4. Contact damping ($\bar{\sigma} = 1000$ s^{-1}) over differently sized regions. The plate was excited at its lower right corner. Left: regions. Right: impulse responses. The dark grey curve is the impulse response with no contact damping applied. The blue and orange curves match the same-colour damping regions in the left-hand side plot.

4.1.2 *Regional Damping*

Similarly, one may apply a non-zero $\bar{\sigma}$ value to a region of the plate, in order to locally effect damping. Figure 4 compares the impulse response of a rectangular plate for two different damping regions with the impulse response of the same plate with no contact damping applied. The expected observation of stronger damping with larger contact area also gives an indication of the requirement of any contact damping having to be distributed in order to facilitate strong damping. In other words, there is little scope for modelling strong damping through a single contact point.

4.2 Dynamic Contact

For dynamic contact, the pressure map $\bar{p}^n_{l,m}$ is updated every sample by linearly interpolating between control-rate updates made every N_b samples, setting $N_\mathrm{b} = 256$. Sound examples are available on the companion webpage [2].

4.2.1 *Damping of a Single Pulse*

A square plate of side length $L_x = L_y = 0.1415$ m and thickness $h = 1.8$ mm is excited at coordinates $x_\mathrm{e} = 0.77L_x$, $y_\mathrm{e} = 0.5L_y$. A velocity output signal is picked up at $x_\mathrm{p} = 0.95L_x$, $y_\mathrm{p} = 0.11L_y$. The plate damping parameters were set as $\sigma_0 = 1$ s^{-1} and $\sigma_2 = 0.001$ m^2 s^{-1}. Contact damping is applied after 1.6 seconds over a circular area of radius 42.4 mm, with the circle centre positioned at $x = 0.1L_x$, $y = 0.5L_y$ (see the top plot in Figure 5). The damping is effected through linearly increasing the control pressure uniformly over the circular region, according to the profile shown in the middle plot of Figure 5. The contact-layer constants featured in equation (6) were set to $a_k = 10000$ N m^{-3}, $a_\sigma = 250$ s^{-1}, and $a_\lambda = 7.2$ kg m^{-2} in order to simulate hand-plate contact with somewhat exaggerated mass loading. The output spectrogram is shown in the bottom plot. Of particular interest is the small decrease in partial frequencies, which is due to the increase in surface density. In addition, it is noticable that some partials are damped more effectively than others, which is mainly due to the chosen region over which damping is applied. One can also observe a small amount of broadband power injected at $t = 1.6$ s, which corresponds to the term

[2] www.socasites.qub.ac.uk/mvanwalstijn/smc21b

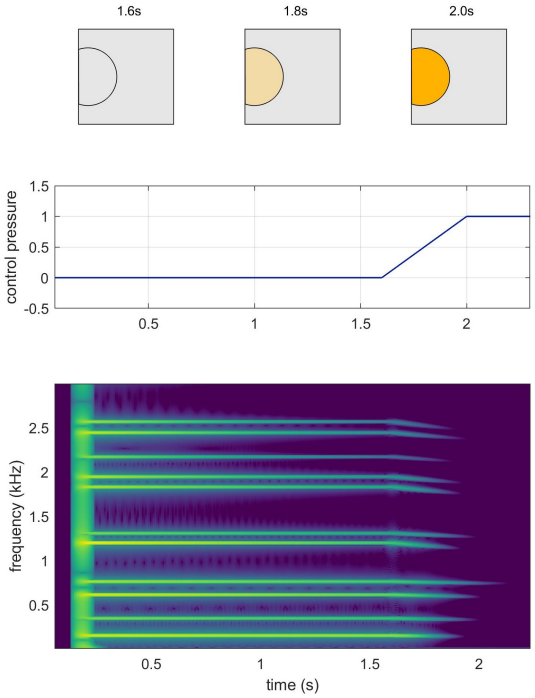


Figure 5. Pulse damping example. The top plot shows the region over which damping was increased according to the profile shown in the middle plot, while the bottom plot shows the output spectrogram.

P_c in the power balance equation in (46), and is due to the variation over time of the contact parameters.

4.2.2 *Linear Water Gong*

In this example, the plate geometrical parameters are set as $L_x = L_y = 0.2983$ m and $h = 2$ mm, and the damping parameters are $\sigma_0 = 1$ s^{-1} and $\sigma_2 = 0.0016$ m^2 s^{-1}. To emulate suspension of the plate, clamping is applied at the corner with coordinates $x = L_x, y = L_y$. The plate is partly submerged into water at the opposite plate corner ($x = 0, y = 0$), the water level reaching 30 percent of the diagonal between the two opposite corners. The plate is excited at $x_\mathrm{e} = 0.88L_x$, $y_\mathrm{e} = 0.87L_y$, and the output velocity is picked up at $x_\mathrm{p} = 0.08L_x$, $y_\mathrm{p} = L_y$. The excitation signal consists of a single short pulse, as shown in the middle plot of Figure 6. During the simulation, the plate is gradually lifted upwards and back downwards (see top plot of Figure 6). Submersion into water is simulated using $a_\lambda = 32$ kg m^{-2} and and setting $\bar{p} = 1$ over the submerged region. The 'additional mass factor' is thus 2, which is in line with theoretical and experimental findings for the first bending mode [9]. The resulting variation of partial frequencies is clearly visible in the spectrogram shown in the bottom plot of Figure 6. Similar to the measured sound shown in Figure 2, the frequency variation differs considerably per partial, which is typical for this type of sound, and sonically distinguishable from phenomena in which all partial frequencies move synchronously.

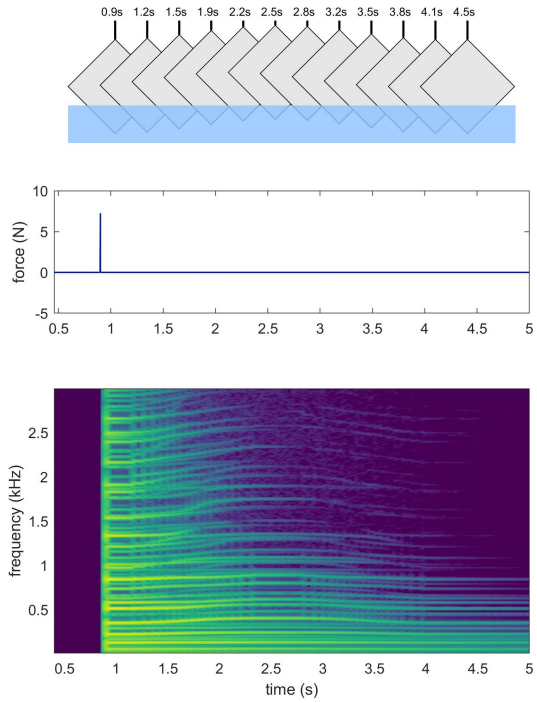


Figure 6. Water gong example. The top plot shows the height position of the suspended plate with respect to the water level. The middle plot shows the excitation signal and the bottom plot shows the output spectrogram.

5. TOWARDS A VIRTUAL-ACOUSTIC PLATE INSTRUMENT

This section outlines ideas and challenges the authors are currently encountering and addressing in applying the methodology towards developing a virtual-acoustic plate instrument.

5.1 Regional vs Global Contact

Recent publications [10, 11] discuss the development of a virtual-acoustic plate prototype instrument, the *Vodhrán*, featuring a modal synthesis algorithm for real-time simulation of plate vibrations, and utilising a Sensel Morph in combination with a contact microphone to sense tactile interaction. One of the ideas behind the design and development of the Vodhrán is achieving and controlling a degree of alignment with real-world counterparts in terms of sonically-relevant affordances.

With the Vodhrán, damping is effected via mapping the overall sum of non-percussive forces detected on the Sensel surface to the plate model's damping parameters. This provides intuitive damping control, but does not account for the specific damping region, and also lacks any additional effects (e.g. mass loading) that accompanies any real-world damping of plates. A real-time implementation of the proposed finite-difference model will allow a more natural, region-dependent form of damping.

One may use the same setup to facilitate and explore clamping configurations. Of particular interest is the potential to reconfigure the plate on the fly not through manipulations in software, but instead through actual clamping of the sensing device, as shown in Figure 7. This will open up ways of exploring the affordances of virtual-acoustic 'prepared plates' entirely through embodied interaction, leveraging skills and embodied knowledge acquired via interaction with real-world plates.

Figure 7. Clamping a Sensel Morph.

Contact that introduces mass-loading over a time-varying plate region, such as in the water gong example, could also be realised in real-time, but - if the affordances of the sensel are to be preserved - this will require an additional sensor interface. One strategy would be to employ a spring-loaded fader (similar to e.g. a synthesizer modulation wheel) to control the 'height' of the plate relative to the water level, as this would provide the musician with a haptic reference. This sensor would then drive the model's surface density via an additional term in the mapping, i.e. changing the first mapping in equation (6) to

$$\bar{\lambda} = a_\lambda \bar{p} + a_\mathrm{w} \bar{p}_\mathrm{w}, \tag{64}$$

where $\bar{p}_\mathrm{w}$ would be set to unity for the submerged region and zero elsewhere, and a_w is an appropriate scalar.

5.2 Computational Challenges

A number of challenges arise in the real-time implementation of the proposed model. Firstly, the implementation in finite-difference form will place limits on the possible size of the plate (in terms of the number of finite-difference nodes). Parallelisation techniques, such as Single Instruction, Multiple Data (SIMD) or Advanced Vector Extensions (AVX) have already proven useful in reducing the relevant CPU time [12,13]. The new, additional challenges that emerge here are due to (a) the need to map the pressure data as sensed on the grid of the sensing device to the finite-difference grid, which involves 2-D (de-)interpolation, and (b) the need to carry out linear interpolation on the contact parameters, which are initially calculated at control rate (i.e. every N_b samples). One investigative route is to establish whether the grid mapping could be done on the sensing device, as such off-loading the main processor.

5.3 Model Extensions

While interesting sounds and articulations are possible with the proposed model, a few further extensions are of

interest. Firstly, most real-world (thin) plates are characterised by strong non-linear behaviour, and this plays an important role in many plates of musical interest. Such behaviour can be simulated by adding a non-linear term to the equation of motion, for example in the form of the von Kármán model [14]; finite-difference discretisations and associated energy analyses developed in [15, 16] could be readily applied to the current model, although with significant additional computational costs. The discretisation of the linear model could also be further investigated, in particular regarding ways to reduce numerical dispersion, possibly using parameterised schemes [17, 18]. Finally, many musical plates are circular, with a free edge; such a boundary could only be realised in the current approach using a staggered approximation of the curved edge. Possibly a better approach would be to apply finite-volume methods at the boundary, as for example employed in [19] for circular plates with fixed and clamped boundary conditions.

6. CONCLUSIONS

The proposed numerical model has been shown to enable simulation of specific, musically-relevant forms of non-excitational contact with a rectangular plate. An underlying motivation of the study is the exploration of new musical behaviours through simultaneous combinations of damping, clamping and mass-loading that would be impractical or expensive with mechanical technology. This will require real-time implementation of the proposed model in conjunction with design and development of sensing strategies, informed by wider instrument design notions. Further future work will focus on model extensions, including nonlinear plate modelling, and on application to membrane-based instruments, which mainly requires a simple replacement in the equation of motion of the stiffness term with a tension term. Of particular interest is the simulation of the tabla, as its performatory vocabulary consists of numerous intricate forms of dynamic contact [20]. Finally, the work presented is also relevant to non-musical sonic interactions, and is part of a wider project [3] that seeks to improve aural immersion in virtual and augmented reality settings.

Acknowledgments

This work was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 812719. We thank the anonymous reviewers whose comments/suggestions helped improve and clarify this manuscript.

7. REFERENCES

[1] P. J. Gomez, "Modern guitar techniques; a view of history, convergence of musical traditions and contemporary works (a guide for composers and guitarists)," Ph.D. dissertation, UC San Diego, 2016.

[2] G. Crumb, "Echoes of Time and the River, Four Processionals for Orchestra (Echoes II)," Edition Peters, 1967.

[3] G. Cuzzucoli and V. Lombardo, "A physical model of the classical guitar, including the player's touch," *Computer Music Journal*, vol. 23, no. 2, pp. 52–69, 1999.

[4] F. Eckerholm and G. Evangelista, "The plucksynth touch string," in *11th Int. Conf. on Digital Audio Effects (DAFx-08)*, 2008.

[5] A. Bhanuprakash, M. van Walstijn, and P. Stapleton, "A finite difference model for articulated slide-string simulation," in *23rd International Conference on Digital Audio Effects (DAFx-20)*, vol. 1, 2020.

[6] S. Bilbao, *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. Wiley Publishing, 2009.

[7] A. Torin, "Percussion instrument modelling in 3d: Sound synthesis through time domain numerical simulation," Ph.D. dissertation, University of Edinburgh, 2015.

[8] K. F. Graff, *Wave motion in elastic solids*, ser. Dover books on physics. New York, NY: Dover, 1991.

[9] M. Haddara and S. Cao, "A study of the dynamic response of submerged rectangular flat plates," *Marine Structures*, vol. 9, no. 10, pp. 913–933, 1996.

[10] L. Pardue, M. Ortiz, M. van Walstijn, P. Stapleton, and R. Matthews, "Vodhrán: collaborative design for evolving a physical model and interface into a proto-instrument," in *International Conference on New Interfaces for Musical Expression*, 2020, pp. 523–524.

[11] M. Rodger, P. Stapleton, M. V. Walstijn, M. Ortiz, and L. Pardue, "What makes a good musical instrument? a matter of processes, ecologies and specificities," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, Birmingham, UK, 2020.

[12] C. Webb and S. Bilbao, "On the Limits of Real-Time Physical Modelling Synthesis with a Modular Environment," in *Proc. of the 18th Int. Conf. on Digital Audio Effects (DAFX-15)*, 2015.

[13] M. van Walstijn and S. Mehes, "An explorative string-bridge-plate model with tunable parameters," in *20th Int. Conf. on Digital Audio Effects (DAFx-17)*, 2017, pp. 291–298.

[14] R. Szilard, *Theory and Analysis of Plates: Classical and Numerical Methods*. Prentice-Hall, 1974.

[15] S. Bilbao, "A family of conservative finite difference schemes for the dynamical von karman plate equations," *Numerical Methods for Partial Differential Equations*, vol. 24, no. 1, pp. 193–216, 2008.

[16] M. Ducceschi, O. Cadot, C. Touzé, and S. Bilbao, "Dynamics of the wave turbulence spectrum in vibrating plates: A numerical investigation using a conservative finite difference scheme," *Physica D: Nonlinear Phenomena*, vol. 280-281, pp. 73 – 85, 2014.

[17] S. Bilbao, L. Savioja, and J. Smith, "Parameterized finite difference schemes for plates: Stability, the reduction of directional dispersion and frequency warping," *IEEE Trans. Audio, Speech, and Lang. Proc.*, vol. 15, no. 4, pp. 1488–1495", 2007.

[18] S. Orr and M. van Walstijn, "Modal representation of the resonant body within a finite difference framework for simulation of string instruments," in *12th Int. Conference on Digital Audio Effects (DAFx-09), Como, Italy*, 2009, pp. 213–220.

[19] B. Hamilton and A. Torin, "Finite difference schemes on hexagonal grids for thin linear plates with finite volume boundaries," in *DAFx*, 2014.

[20] A. Kapur, G. Essl, P. Davidson, and P. R. Cook, "The electronic tabla controller," *Journal of New Music Research*, vol. 32, no. 4, pp. 351–359, 2003.

[3] https://vrace-etn.eu/

REAL-TIME IMPLEMENTATION OF THE SHAMISEN USING FINITE DIFFERENCE SCHEMES

Titas LASICKAS (tlasic16@student.aau.dk) [1], **Silvin WILLEMSEN** (sil@create.aau.dk) [2], and **Stefania SERAFIN** (sts@create.aau.dk) [2]

[1] *CREATE*, **Aalborg University**, Copenhagen, Denmark
[2] *Multisensory Experience Lab, CREATE*, **Aalborg University**, Copenhagen, Denmark

ABSTRACT

The shamisen is a Japanese three-stringed lute. It is a chordophone that has the front of the body covered by a tensioned membrane which greatly contributes to the distinct sound of the instrument. Although the shamisen is a traditional Japanese instrument, it is a rare instrument in the rest of the world, making it mostly inaccessible by the majority of artists. To our knowledge, no physically modelled synthesizer of the shamisen is available, forcing producers and musicians to use samples. The objective of this paper is to make the shamisen's distinct sound more accessible to digital music artists. The real-time implementation of the shamisen physical model is presented along with the derivation of solution using the finite-difference time-domain (FDTD) methods. The digital instrument sounds mostly as intended, though lacking the shamisen's distinct buzzing sound requiring further development.

1. INTRODUCTION

The shamisen (see Figure 1) is a Japanese three-stringed lute, with origins in China. This instrument is a chordophone that has a membrane covering its soundbox; this stretched membrane contributes to the distinct sound of the instrument. The instrument is played using a bachi, a large plectrum which is held in one hand, while using the fingers of the other hand to pinch the strings against the neck of the instrument, allowing the user to play different pitches. The timbre of the shamisen has a very distinct buzzing which is associated with a low nut which lets a vibrating string come in the contact with the neck [1] and the specific playing technique that increases the percussiveness of the instrument by hitting the body with the bachi during the plucking of the strings [2].

The main goal of this paper is the digitalization and real-time simulation of an instrument that, due to its rarity, is not easily available to the majority of artists. Currently, the only method of obtaining shamisen sounds without having the instrument at hand is by using a sample of the instrument [4] or one of the audio plugins [5–7] which are also based on a sampled shamisen. Usually, a single sample of a note played at a single pitch is pitch-shifted enabling artist to play melodies and chords. Such a method relies on the recordings being done in an anechoic chamber to reduce the effects of the room response. Moreover, due to only usually recorded one note sample, the pitch shifting can introduce artifacts. Even when the sampling is done for all possible pitches on each string, the user is stuck with the way the performer played the instrument during the recording. To rectify these undesirable qualities of the sampling method and to create more scope for skillful articulation when performing, the shamisen can be simulated using a physical model instead.

Figure 1. The shamisen (taken from [3]).

Although several stringed musical instruments have been mostly simulated using digital waveguides [8–10], in this paper we model the shamisen by using Finite Difference Time Domain (FDTD) methods [11]. FDTD methods require developing a full mathematical description of the system. Such a description development uses partial differential equations which are discretized using FDTD methods, yielding finite difference schemes (FDSs). FDTD methods provide better spatial accuracy when the model has frequency-dependent damping and dispersion [12, 13]; in addition, FDTD methods are more flexible as no assumptions are being made about the linearity of the solution [11]. Alternatively, a modal approach – such as in [14, 15] – could be used as it is generally much more efficient. Additionally, modal synthesis for 2D systems appeared in [16, 17], however, to retain generality for control and easier implementation when connecting multiple models, FDTD methods are chosen here . In addition, the nonlinear collision of the bachi with the membrane can be modelled straight-forwardly using FDTD methods. The real-time implementation of similar models, modelled using FDTD approach have been achieved by the authors of [18], where a real-time banjo is recreated using a field programmable gate array and by the authors of [19] where

a real-time implementation of tromba marina is achieved using C++ and the JUCE framework [20]. In this work, we offer a real-time implementation of a physical model of the shamisen based on FDTD methods due to its benefits regarding flexibility and accuracy.

This paper is organized as follows: Section 2 presents the physical models used to model the shamisen and Section 3 shows the discretization of these models. Section 4 shows details of the implementation such as parameter choices, excitation and the output of the system, a high level overview of the calculation order and the graphical user interface and control. Section 5 shows the results regarding output sound and speed of the algorithm and discusses these, while the future work as well as concluding remarks are presented in Section 6.

2. MODELING THE SYSTEM

The shamisen can be simplified down to five major parts that make up the whole instrument: the three strings, the bridge and the membrane. The rest of the body is excluded to reduce model complexity which is desirable in order to achieve a model capable of running in real-time. This section will derive partial differential equations (PDEs) in a form

$$\mathcal{L}q = 0, \tag{1}$$

for the mentioned parts of the instrument in isolation. Here, $\mathcal{L}$ is a partial differential operator and $q = q(\boldsymbol{x}, t)$ with time t and spatial coordinate $\boldsymbol{x} \in \mathcal{D}$, where domain $\mathcal{D}$ is one-dimensional for the strings and the bridge, and two-dimensional for the membrane. Moreover, the subscripts 's', 'b' and 'm' indicate that the variables are used for the strings, bridge and the membrane respectively.

2.1 Damped Stiff String

Consider a damped stiff string of length L_s defined over domain $\mathcal{D} = \mathcal{D}_\mathrm{s} = [0, L_\mathrm{s}]$. Recalling Equation (1), the dependent variable $q = u(x, t)$ describes the transverse displacement of the string. Furthermore, the operator $\mathcal{L} = \mathcal{L}_\mathrm{s}$ can be defined as

$$\begin{aligned}\mathcal{L}_\mathrm{s} = \rho_\mathrm{s} A_\mathrm{s} \partial_t^2 - T_\mathrm{s} \partial_x^2 + E_\mathrm{s} I_\mathrm{s} \partial_x^4 \\ + 2\rho_\mathrm{s} A_\mathrm{s} \sigma_{0,\mathrm{s}} \partial_t - 2\rho_\mathrm{s} A_\mathrm{s} \sigma_{1,\mathrm{s}} \partial_t \partial_x^2.\end{aligned} \tag{2}$$

Here, ∂_t and ∂_x indicate a partial differentiation with respect to time and space respectively. Furthermore, various parameters are used to define the string behaviour such as the material density ρ_s, (circular) cross sectional area $A_\mathrm{s} = \pi r^2$, r being the radius of the string, tension of the string T_s, Young's modulus of the string material E_s and area moment inertia $I_\mathrm{s} = \pi r^4/4$, along with damping coefficients $\sigma_{0,\mathrm{s}}$ and $\sigma_{1,\mathrm{s}}$.

2.2 Damped Bridge

The bridge of the shamisen is modeled as a damped linear bar of length L_b with domain $\mathcal{D}_\mathrm{b} = [0, L_\mathrm{b}]$, and dependent variable $q = v(x, t)$ describing the transverse displacement. The operator $\mathcal{L} = \mathcal{L}_\mathrm{b}$ is similar to $\mathcal{L}_\mathrm{b}$ in Equation (2) without the tension term resulting in the following

$$\begin{aligned}\mathcal{L}_\mathrm{b} = \rho_\mathrm{b} A_\mathrm{b} \partial_t^2 + E_\mathrm{b} I_\mathrm{b} \partial_x^4 \\ + 2\rho_\mathrm{b} A_\mathrm{b} \sigma_{0,\mathrm{b}} \partial_t - 2\rho_\mathrm{b} A_\mathrm{b} \sigma_{1,\mathrm{b}} \partial_t \partial_x^2.\end{aligned} \tag{3}$$

Various parameters are used to define the behaviour of the bridge, such as the material density ρ_b, (rectangular) cross sectional area $A_\mathrm{b} = bH_\mathrm{b}$, b being the width and H_b the thickness of the bridge, the Young's modulus of the bridge material E_b and the area moment inertia $I_\mathrm{b} = \frac{1}{12} b H_\mathrm{b}^3$, along with damping coefficients $\sigma_{0,\mathrm{b}}$ and $\sigma_{1,\mathrm{b}}$.

2.3 Damped stiff membrane

Finally, the membrane covering the instrument's soundbox is modelled as a rectangular damped stiff membrane with side lengths L_x and L_y, domain $\mathcal{D} = \mathcal{D}_\mathrm{m} = [0, L_x] \times [0, L_y]$ and dependent variable $q = w(x, y, t)$. The stiffness of the membrane is simulated using a Kirchhoff thin plate stiffness term. Using the 2D Laplacian

$$\Delta \triangleq \partial_x^2 + \partial_y^2, \tag{4}$$

where ∂_x and ∂_y indicate partial differentiation with respect to two spatial dimensions, the operator $\mathcal{L} = \mathcal{L}_\mathrm{m}$ can be defined as:

$$\begin{aligned}\mathcal{L}_\mathrm{m} = \rho_\mathrm{m} H \partial_t^2 - T_\mathrm{m} \Delta + D \Delta\Delta \\ + 2\rho_\mathrm{m} H \sigma_{0,\mathrm{m}} \partial_t - 2\rho_\mathrm{m} H \sigma_{1,\mathrm{m}} \partial_t \Delta.\end{aligned} \tag{5}$$

Again, various parameters are used to define the behaviour of the membrane, such as the material density ρ_m, the membrane thickness H_m, the tension of the membrane T_m, the stiffness parameter $D = E_\mathrm{m} H_\mathrm{m}^3 / 12(1 - \nu^2)$, where E_m is the Young's modulus of the membrane material and Poisson ratio ν, along with the damping coefficients $\sigma_{0,\mathrm{m}}$ and $\sigma_{1,\mathrm{m}}$.

2.4 Boundary Conditions

Something about distributed systems requiring definitions for what happens at the boundaries.

The string is clamped at the boundaries according to

$$u = \partial_x u = 0, \quad \text{where} \quad x = 0, L_\mathrm{s}. \tag{6}$$

Similarly, the membrane is clamped according to

$$w = \mathbf{n} \cdot \nabla w = 0 \tag{7}$$

where ∇w denotes the gradient of w and $\mathbf{n}$ is a normal to the membrane area at the boundary.

The bridge is free at the boundaries according to

$$\partial_x^2 v = \partial_x^3 v = 0 \quad \text{where} \qquad x = 0, L_\mathrm{b}. \tag{8}$$

2.5 The complete system

Until now, only systems in isolation have been considered, i.e., of form (1). To connect the different components, a spatial Dirac delta function $\delta(\boldsymbol{x} - \boldsymbol{x}_\mathrm{c})$ can be used,

which locates the interaction force between two components to the location $\boldsymbol{x}_{\mathrm{c}} \in \mathcal{D}$. The complete system for the shamisen can be written as

$$\begin{cases} \mathcal{L}_{si} u_i &= -\delta(x - x_{si}) F_{\mathrm{t}i}, & \text{(9a)} \\ \mathcal{L}_{\mathrm{b}} v &= \sum_{i=1}^{3} \delta(x - x_{\mathrm{b}i}) F_{\mathrm{t}i} & \text{(9b)} \\ & \quad - \delta(x - x_{\mathrm{bL}}) F_{\mathrm{bL}} - \delta(x - x_{\mathrm{bR}}) F_{\mathrm{bR}}, \\ \mathcal{L}_{\mathrm{m}} w &= \delta(x - x_{\mathrm{mL}}, y - y_{\mathrm{mL}}) F_{\mathrm{bL}} & \text{(9c)} \\ & \quad + \delta(x - x_{\mathrm{mR}}, y - y_{\mathrm{mR}}) F_{\mathrm{bR}}, \end{cases}$$

where subscript i indicates the i'th string, $F_{\mathrm{t}i}$ is the force between individual strings and the bridge and F_{bL} and F_{bR} are the forces between the bridge and the membrane at the left (L) and right (R) sides of the bridge respectively. The locations x_{si} and $x_{\mathrm{b}i}$ are the locations where the bridge connects to each individual string and vice versa, the 's' subscript denotes that it is a location along a string and the 'b' subscript denotes the location on the bridge. The three strings have one connection each, while the bridge is connected to all three strings. In addition, the locations with subscripts 'bL' and 'mL' correspond to where the membrane connects to the left side of the bridge and vice versa. The same is indicated for the right side using subscripts 'bR' and 'mR'.

3. DISCRETIZATION

This section discretizes the full system described in (9) using FDTD methods. These methods subdivide continuous systems (such as described in the previous section) into time samples and grid points in space. The notation used in this section follows [11].

3.1 Finite Difference Operators

The first step of implementing FDSs is to define a sampling interval of the continuous system. Time is discretized as $t = nk$, with temporal index $n \in [0, 1, 2...\infty)$ and sampling interval $k = 1/f_{\mathrm{s}}$ where f_{s} is the sample rate. Space is discretized as $\boldsymbol{x} = \boldsymbol{l}h$, where the dimension of spatial index $\boldsymbol{l}$ is determined by the number of dimensions of the system at hand and h is some spatial sampling interval. Since up-sampling or down-sampling is not needed, k remains the same for all the parts of the model. Once the sampling intervals have been defined, $q(\boldsymbol{x}, t)$ is approximated as a grid function $q_{\boldsymbol{l}}^n$.

Regardless of the dimension of $\boldsymbol{l}$, shift operators can be applied to a grid function. Temporal shift operators are defined as

$$e_{t+} q_{\boldsymbol{l}}^n = q_{\boldsymbol{l}}^{n+1}, \qquad e_{t-} q_{\boldsymbol{l}}^n = q_{\boldsymbol{l}}^{n-1}. \tag{10}$$

Using these shift operators, more complex difference operators can be defined for approximating a first-order time derivative. The forward, backward and centered difference in time operators can be defined as

$$\delta_{t+} := \frac{e_{t+} - 1}{k} \approx \partial_t, \quad \delta_{t-} := \frac{1 - e_{t-}}{k} \approx \partial_t, \quad \delta_{t\cdot} := \frac{e_{t+} - e_{t-}}{2k} \approx \partial_t. \tag{11}$$

where the forward and backward difference are first-order accurate and the centered difference is second-order accurate. Using these first-order difference operators, the second-order difference operator can be defined as a combination of the forward and backward difference operators:

$$\delta_{tt} = \delta_{t+}\delta_{t-} := \frac{e_{t+} - 2 + e_{t-}}{k^2} \approx \partial_t^2. \tag{12}$$

Spatial shift operators in 1D, i.e., $\boldsymbol{l} = l$ can be similarly defined as

$$e_{x+} q_l^n = q_{l+1}^n, \qquad e_{x-} q_l^n = q_{l-1}^n, \tag{13}$$

after which the following first-order difference operators can be defined using the spatial sampling interval h

$$\delta_{x+} := \frac{e_{x+} - 1}{h} \approx \partial_x, \quad \delta_{x-} := \frac{1 - e_{x-}}{h} \approx \partial_x, \quad \delta_{x\cdot} := \frac{e_{x+} - e_{x-}}{2h} \approx \partial_x, \tag{14}$$

(following the same orders of accuracy as the first-order time operators) and second-order difference in space operator

$$\delta_{xx} = \delta_{x+}\delta_{x-} := \frac{e_{x+} - 2 + e_{x-}}{h^2} \approx \partial_x^2. \tag{15}$$

The fourth-order spatial derivative operator is obtained by applying operator (15) twice:

$$\delta_{xxxx} = \delta_{xx}\delta_{xx} = \frac{1}{h^4}(e_{t+}^2 - 4e_{t+} + 6 - 4e_{t-} + e_{t-}^2), \tag{16}$$

where a squared shift operator simply means to apply it twice.

The mixed temporal-spatial derivative operator is obtained by applying the backward-time difference operator from (10) to operator (15)

$$\delta_{t-}\delta_{xx} := \frac{e_{x+} - 2 + e_{x-} - e_{t-}(e_{x+} - 2 + e_{x-})}{kh^2} \approx \partial_t \partial_x^2, \tag{17}$$

where a backward difference in time is chosen to keep the system explicit.

Furthermore, in 2D, i.e., $\boldsymbol{l} = l, m$, shift operators are defined as

$$\begin{aligned} e_{x+} q_{l,m}^n &= q_{l+1,m}^n, \quad & e_{x-} q_{l,m}^n &= q_{l-1,m}^n, \\ e_{y+} q_{l,m}^n &= q_{l,m+1}^n, \quad & e_{y-} q_{l,m}^n &= q_{l,m-1}^n, \end{aligned} \tag{18}$$

and finite difference operators can be defined as

$$\begin{aligned} \delta_\Delta = \delta_{xx} + \delta_{yy} \approx \Delta \quad \text{and} \\ \delta_\Delta \delta_\Delta = \delta_{xxxx} + 2\delta_{xx}\delta_{yy} + \delta_{yyyy} \approx \Delta\Delta, \end{aligned} \tag{19}$$

where δ_{yy} and δ_{yyyy} are similarly defined to Equations (15) and (16) but using shifting operator e_{y+}. Finally, the mixed temporal-spatial operator in 2D is similarly defined as operator (17) using a backward difference in time operator.

It is important to mention that the discrete FDSs derived from the continuous equations such as (9) are an approximation rather than a sampled version of the system.

3.2 Discrete Models

In the following, parameters containing a subscript i indicate that these vary between each individual string.

In the case of the strings, we use $x = lh_{\mathrm{s}i}$ to get $u_i(x,t) \approx u_{i,l}^n$, where $l \in [0,\ldots,N_{\mathrm{s}i}]$ and $N_{\mathrm{s}i} = \mathrm{floor}(L_{\mathrm{s}}/h_{\mathrm{s}i})$. All the strings are the same length, L_{s}, but the grid spacing $h_{\mathrm{s}i}$ depends on the individual string parameters. The minimal grid spacing where the solution is stable is calculated using von Neumann stability analysis [11]:

$$h_{\mathrm{s}i} \geq \sqrt{\frac{c_{\mathrm{s}i}^2 k^2 + 4\sigma_{1,\mathrm{s}}k + \sqrt{(c_{\mathrm{s}i}^2 k^2 + 4\sigma_{1,\mathrm{s}}k)^2 + 16\kappa_{\mathrm{s}i}^2 k^2}}{2}}. \tag{20}$$

Here, the wave speed $c_{\mathrm{s}i} = \sqrt{T_{\mathrm{s}i}/\rho_{\mathrm{s}}A_{\mathrm{s}i}}$ and the stiffness $\kappa_{\mathrm{s}i} = \sqrt{E_{\mathrm{s}}I_{\mathrm{s}i}/\rho_{\mathrm{s}}A_{\mathrm{s}i}}$. The closer $h_{\mathrm{s}i}$ is to this condition, the higher the simulation quality will be. The same goes for the conditions given below.

For the bridge we use $x = lh_{\mathrm{b}}$ to get $v(x,t) \approx v_l^n$, where $l \in [0,\ldots,N_{\mathrm{b}}]$ and $N_{\mathrm{b}} = \mathrm{floor}(L_{\mathrm{b}}/h_{\mathrm{b}})$. The grid spacing h_{b} is defined as

$$h_{\mathrm{b}} \geq \sqrt{2k\left(\sigma_{1,\mathrm{b}} + \sqrt{\sigma_{1,\mathrm{b}}^2 + \kappa_{\mathrm{b}}^2}\right)}, \tag{21}$$

with stiffness parameter $\kappa_{\mathrm{b}} = \sqrt{E_{\mathrm{b}}{H_{\mathrm{b}}}^2/12\rho_{\mathrm{b}}}$.

In the case of the membrane, we use $x = lh_{\mathrm{m}}$ and $y = mh_{\mathrm{m}}$ to get $w(x,y,t) \approx w_{l,m}^n$ where the horizontal index $l \in [0,\ldots,N_x]$ with $N_x = \mathrm{floor}(L_x/h_{\mathrm{m}})$ and vertical index $m \in [0,\ldots,N_y]$ with $N_y = \mathrm{floor}(L_y/h_{\mathrm{m}})$. The membrane is modelled to be square, i.e., $L_x = L_y$, making $N_x = N_y$. The minimal grid spacing h_{m} is calculated using

$$h_{\mathrm{m}} \geq \sqrt{c_{\mathrm{m}}^2 k^2 + 4\sigma_{1,\mathrm{m}}k + \sqrt{(c_{\mathrm{m}}^2 k^2 + 4\sigma_{1,\mathrm{m}}k)^2 + 16\kappa_{\mathrm{m}}^2 k^2}}, \tag{22}$$

where the membrane wave speed $c_{\mathrm{m}} = \sqrt{T_{\mathrm{m}}/\rho_{\mathrm{m}}H}$ and the stiffness $\kappa_{\mathrm{m}} = \sqrt{D/\rho_{\mathrm{m}}H}$.

In order to discretize the Dirac delta functions found in system (9) we introduce the spreading operator $J(\boldsymbol{x}_{\mathrm{c}})$ that applies the force to the coordinate $\boldsymbol{x}_{\mathrm{c}}$, which is defined as [19]

$$J(\boldsymbol{x}_{\mathrm{c}}) = \begin{cases} \frac{1}{h^d}, & \boldsymbol{l} = \boldsymbol{l}_{\mathrm{c}} = \mathrm{round}(\boldsymbol{x}_{\mathrm{c}}/h), \\ 0, & \text{otherwise.} \end{cases} \tag{23}$$

The rounding function here is used for simplicity, but higher-order spreading functions as found in [11] may be used. In the equation (23) d is a number of the dimensions of domain $\mathcal{D}$ on which $\boldsymbol{x}_{\mathrm{c}}$ is defined. So in the case of the strings and the bridge $d = 1$ and $d = 2$ for the membrane.

3.3 Boundary Conditions

Recalling N_{s}, N_{b}, N_x and N_y from Section 3.2, and the finite difference operators from Section 3.1, the strings and the membrane have their boundaries set as clamped which yields boundary conditions for the strings to be

$$u_0^n = \delta_{x+}u_0^n = 0, \quad \text{and} \quad u_{N_{\mathrm{s}}}^n = \delta_{x-}u_{N_{\mathrm{s}}}^n = 0, \tag{24}$$

and the membrane to be

$$\begin{aligned} w_{0,m}^n &= \delta_{x+}w_{0,m}^n = 0, \\ w_{N_x,m}^n &= \delta_{x-}w_{N_x,m}^n = 0, \\ w_{l,0}^n &= \delta_{y+}w_{l,0}^n = 0, \\ w_{l,N_y}^n &= \delta_{y-}w_{l,N_y}^n = 0. \end{aligned} \tag{25}$$

The bridge on the other hand, has a free boundary condition, which is described as

$$\begin{aligned} \delta_{xx}v_0^n = \delta_{xx}\delta_{x\cdot}v_0^n = 0, \quad \text{and} \\ \delta_{xx}v_{N_{\mathrm{b}}}^n = \delta_{xx}\delta_{x\cdot}v_{N_{\mathrm{b}}}^n = 0. \end{aligned} \tag{26}$$

3.4 Complete Discrete System

The discretized version of continuous complete system (9) is

$$\begin{cases} \ell_{\mathrm{s}i}u_{i,l}^n &= -J(x_{\mathrm{s}i})F_{\mathrm{t}i}, & \text{(27a)} \\ \ell_{\mathrm{b}}v_l^n &= \sum_{i=1}^{3} J(x_{\mathrm{b}i})F_{\mathrm{t}i} - J(x_{\mathrm{bL}})F_{\mathrm{bL}} & \text{(27b)} \\ &\quad - J(x_{\mathrm{bR}})F_{\mathrm{bR}}, \\ \ell_{\mathrm{m}}w_{l,m}^n &= J(x_{\mathrm{mL}},y_{\mathrm{mL}})F_{\mathrm{bL}} & \text{(27c)} \\ &\quad + J(x_{\mathrm{mR}},y_{\mathrm{mR}})F_{\mathrm{bR}}, \end{cases}$$

where the ℓ operators are discretized versions of the partial differential operators $\mathcal{L}$ in system (9) and follow [11] using centered temporal differences for the frequency independent damping terms and backward differences for the frequency dependent damping terms to keep the system explicit (as mentioned in Section (3.1)). Due to the fact that the forces acting on two connected components are equal and opposite due to the rigid connections used, it can be shown that the system is stable as a whole [11].

3.5 Coupling

The shamisen model consists of 3 strings, a bridge and a membrane, all coupled together to form a complete instrument (see Figure 2).

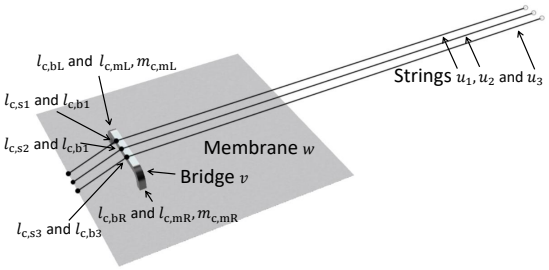


Figure 2. Schematic showing the full coupled system. The different components are highlighted, together with the locations at which they are connected. For the latter, also see Table 1.

The only thing left to do is to calculate the interaction forces between the components. To do this, all schemes in system (27) need to be expanded (written in full) around

every individual connection location which is done by taking an inner product with every individual spreading operator J [11]. Expanding ℓ_{si} in Equation (27a) and using subscript $\mathcal{S}_i = i, l_{\mathrm{c},si}$ for compactness, where $i \in [1, 2, 3]$ is the string index and $l_{\mathrm{c},si}$ is the corresponding location on string i where it connects to the bridge, we obtain the following

$$\begin{aligned}\delta_{tt} u^n_{\mathcal{S}_i} = c^2_{si}\delta_{xx} u^n_{\mathcal{S}_i} - \kappa^2_{si}\delta_{xxxx} u^n_{\mathcal{S}_i} - 2\sigma_{0,\mathrm{s}}\delta_{t\cdot} u^n_{\mathcal{S}_i} \\ + 2\sigma_{1,\mathrm{s}}\delta_{t-}\delta_{xx} u^n_{\mathcal{S}_i} - \frac{F_{\mathrm{t}i}}{h_{si}\rho_{\mathrm{s}} A_{si}}\,.\end{aligned} \tag{28}$$

Similarly, for the bridge, we can expand ℓ_{b} from Equation (27b) and take the inner product with the spreading operators belonging to the connections with the strings to obtain

$$\begin{aligned}\delta_{tt} v^n_{l_{ci,\mathrm{b}}} = &-\kappa^2_{\mathrm{b}}\delta_{xxxx} v^n_{l_{ci,\mathrm{b}}} - 2\sigma_{0,\mathrm{b}}\delta_{t\cdot} v^n_{l_{ci,\mathrm{b}}} \\ &+ 2\sigma_{1,\mathrm{b}}\delta_{t-}\delta_{xx} v^n_{l_{ci,\mathrm{b}}} + \frac{F_{\mathrm{t}i}}{h_{\mathrm{b}}\rho_{\mathrm{b}} A_{\mathrm{b}}},\end{aligned} \tag{29}$$

where $l_{ci,\mathrm{b}}$ indicates the location where the bridge connects to string i. Taking the inner product with the spreading operators belonging to the connections with the membrane yields

$$\begin{aligned}\delta_{tt} v^n_{l_{\mathrm{c,bL}}} = &-\kappa^2_{\mathrm{b}}\delta_{xxxx} v^n_{l_{\mathrm{c,bL}}} - 2\sigma_{0,\mathrm{b}}\delta_{t\cdot} v^n_{l_{\mathrm{c,bL}}} \\ &+ 2\sigma_{1,\mathrm{b}}\delta_{t-}\delta_{xx} v^n_{l_{\mathrm{c,bL}}} - \frac{F_{\mathrm{bL}}}{h_{\mathrm{b}}\rho_{\mathrm{b}} A_{\mathrm{b}}}, \\ \delta_{tt} v^n_{l_{\mathrm{c,bR}}} = &-\kappa^2_{\mathrm{b}}\delta_{xxxx} v^n_{l_{\mathrm{b,bR}}} - 2\sigma_{0,\mathrm{b}}\delta_{t\cdot} v^n_{l_{\mathrm{c,bR}}} \\ &+ 2\sigma_{1,\mathrm{b}}\delta_{t-}\delta_{xx} v^n_{l_{\mathrm{c,bR}}} - \frac{F_{\mathrm{bR}}}{h_{\mathrm{b}}\rho_{\mathrm{b}} A_{\mathrm{b}}},\end{aligned} \tag{30}$$

where $l_{\mathrm{c,bL}}$ and $l_{\mathrm{c,bR}}$ indicate the locations where the left and right side of bridge connect to the membrane respectively.

Finally, we can follow the same process for the membrane. For brevity, the locations where the bridge and the membrane are connected are defined as $\mathrm{mL} = l_{\mathrm{c,mL}}, m_{\mathrm{c,mL}}$ for the left connection location and $\mathrm{mR} = l_{\mathrm{c,mR}}, m_{\mathrm{c,mR}}$ for the right:

$$\begin{aligned}&\delta_{tt} w^n_{\mathrm{mL}} = c^2_{\mathrm{m}}\delta_{\Delta} w^n_{\mathrm{mL}} - \kappa^2_{\mathrm{m}}\delta_{\Delta}\delta_{\Delta} w^n_{\mathrm{mL}} \\ &- 2\sigma_{0,\mathrm{m}}\delta_{t\cdot} w^n_{\mathrm{mL}} + 2\sigma_{1,\mathrm{m}}\delta_{t-}\delta_{\Delta} w^n_{\mathrm{mL}} + \frac{F_{\mathrm{bL}}}{h^2_{\mathrm{m}}\rho_{\mathrm{m}} H_{\mathrm{m}}}, \\ &\delta_{tt} w^n_{\mathrm{mR}} = c^2_{\mathrm{m}}\delta_{\Delta} w^n_{\mathrm{mR}} - \kappa^2_{\mathrm{m}}\delta_{\Delta}\delta_{\Delta} w^n_{\mathrm{mR}} \\ &- 2\sigma_{0,\mathrm{m}}\delta_{t\cdot} w^n_{\mathrm{mR}} + 2\sigma_{1,\mathrm{m}}\delta_{t-}\delta_{\Delta} w^n_{\mathrm{mR}} + \frac{F_{\mathrm{bR}}}{h^2_{\mathrm{m}}\rho_{\mathrm{m}} H_{\mathrm{m}}}.\end{aligned} \tag{31}$$

All the connection locations where the individual components are coupled together are summarised in Table 1.

There are multiple ways of connecting components together [11]. In this work, rigid connections are assumed, which means that for all n

$$\begin{aligned}u^n_{\mathcal{S}_i} = v^n_{l_{ci,\mathrm{b}}}, \quad v^n_{l_{\mathrm{c,bL}}} = w^n_{\mathrm{mL}}, \\ \text{and} \quad v^n_{l_{\mathrm{c,bR}}} = w^n_{\mathrm{mR}}\,.\end{aligned} \tag{32}$$

All the operators in Equations (28), (29) and (30) can be expanded according to the definitions given in Section (3.1) and solved for the states at $n+1$. We can introduce an intermediate state q^{I}_l which is q^{n+1}_l without the effect of the connection forces. As we know that Equation (32) is true for all n, and thus also for $n+1$, we can set the expanded schemes at their connection locations at $n+1$ equal to each other. This yields, for the string-bridge connections

$$u^{\mathrm{I}}_{\mathcal{S}_i} - \frac{F_{\mathrm{t}i}k^2}{h_{si}\rho_{\mathrm{s}} A_{si}(\sigma_{0,\mathrm{s}}k+1)} = v^{\mathrm{I}}_{l_{ci,\mathrm{b}}} + \frac{F_{\mathrm{t}i}k^2}{h_{\mathrm{b}}\rho_{\mathrm{b}} A_{\mathrm{b}}(\sigma_{0,\mathrm{b}}k+1)}, \tag{33}$$

and for the bridge-membrane connections

$$\begin{aligned}v^{\mathrm{I}}_{l_{\mathrm{c,bL}}} - \frac{F_{\mathrm{bL}}k^2}{h_{\mathrm{b}}\rho_{\mathrm{b}} A_{\mathrm{b}}(\sigma_{0,\mathrm{b}}k+1)} = w^{\mathrm{I}}_{\mathrm{mL}} + \frac{F_{\mathrm{bL}}k^2}{h^2_{\mathrm{m}}\rho_{\mathrm{m}} H_{\mathrm{m}}(\sigma_{0,\mathrm{m}}k+1)}, \\ v^{\mathrm{I}}_{l_{\mathrm{c,bR}}} - \frac{F_{\mathrm{bR}}k^2}{h_{\mathrm{b}}\rho_{\mathrm{b}} A_{\mathrm{b}}(\sigma_{0,\mathrm{b}}k+1)} = w^{\mathrm{I}}_{\mathrm{mR}} + \frac{F_{\mathrm{bR}}k^2}{h^2_{\mathrm{m}}\rho_{\mathrm{m}} H_{\mathrm{m}}(\sigma_{0,\mathrm{m}}k+1)}.\end{aligned} \tag{34}$$

These can then be solved for the forces which can finally be substituted back into system (27).

Strings	$l_{\mathrm{c,s1}}$	$l_{\mathrm{c,s2}}$	$l_{\mathrm{c,s3}}$
Bridge	$l_{\mathrm{c1,b}}$ $l_{\mathrm{c,bL}}$	$l_{\mathrm{c2,b}}$ $l_{\mathrm{c,bR}}$	$l_{\mathrm{c3,b}}$
Membrane	$l_{\mathrm{c,mL}}$, $m_{\mathrm{c,mL}}$	$l_{\mathrm{c,mR}}$, $m_{\mathrm{c,mR}}$	

Table 1. All the connection locations in discrete system (27) (also see Figure 2). Subscripts 'c, s1', 'c, s2' and 'c, s3' indicate the connection points to the bridge on the different strings, 'c1, b', 'c2, b' and 'c3, b' are the corresponding connection points on the bridge for these strings. The subscripts'c, bL' and 'c, bR' indicate connection points on the left the right side of the bridge, in turn, 'c, mL' and 'c, mR' are the same points on the membrane.

4. IMPLEMENTATION

The physical model of the shamisen has been implemented in real-time in C++ using the JUCE framework [20]. The code is available at [25]. The control of the digital instrument is limited to the excitation of the separate parts of the instrument including the bridge and the membrane. This section will elaborate on some important considerations regarding the setup of the system, the algorithm and the parameter design. In the end, the graphical user interface (GUI) of the application will be presented.

4.1 System setup

The parameters are set at the beginning of the simulation according to Table 2. From this, the spatial sampling intervals h for each individual component are calculated using conditions (20), (21) and (22). After h_{m} is calculated, we check whether it is smaller than a set minimum value and if it is, use this value instead. Though reducing the quality of the membrane simulation, it increases the computational speed, ultimately allowing the real-time implementation to run smoothly. The value $h_{\mathrm{m,min}} = 0.03$ was heuristically found to be a good trade off between speed and quality.

The spatial sampling intervals are then used to calculate the number of grid points $N_{si}, N_{\mathrm{b}}, N_x$ and N_y which determine the sizes of the state vectors of each component. For every component three vectors (or matrices in the case of the membrane) need to be initialised, saving the states of

Symbol	Value (Unit)	Parameter
b	$2.69 \cdot 10^{-3}$ (m)*	Width of the bridge
E_s	$9.9 \cdot 10^{9}$ (Pa)**	String Young's Mod.
E_b	$9.5 \cdot 10^{9}$ (Pa)*	Bridge Young's Mod.
E_m	$3 \cdot 10^{9}$ (Pa)*	Memb. Young's Mod.
H_b	$7.5 \cdot 10^{-}3$ (m)*	Bridge thickness
H_m	$0.2 \cdot 10^{-}3$ (m)**	Membrane thickness
L_s	1 (m)†	String length
L_b	1 (m)†	Bridge length
L_x	1 (m)†	Membrane length
L_y	1 (m)†	Membrane width
ν	0.4	Poisson's ratio
$r_{\mathrm{s}1}$	$4.15 \cdot 10^{-4}$ (m)	Radius string 1
$r_{\mathrm{s}2}$	$2.83 \cdot 10^{-4}$ (m)	Radius string 2
$r_{\mathrm{s}3}$	$2.10 \cdot 10^{-4}$ (m)	Radius string 3
ρ_s	1156 (kg/m^3)**	String density
ρ_b	500 (kg/m^3)*	Bridge density
ρ_m	1150 (kg/m^3)*	Memb. density
		Frequency independent damping
$\sigma_{0,\mathrm{s}}$	1.378 (s^{-1})	String damping
$\sigma_{0,\mathrm{b}}$	1.343 (s^{-1})	Bridge damping
$\sigma_{0,\mathrm{m}}$	2.756 (s^{-1})	Membrane damping
		Frequency dependent damping
$\sigma_{1,\mathrm{s}}$	$3.57 \cdot 10^{-3}$ (m^2/s)	String damping
$\sigma_{1,\mathrm{b}}$	$7.59 \cdot 10^{-2}$ (m^2/s)	Bridge damping
$\sigma_{1,\mathrm{m}}$	0.192 (m^2/s)	Membrane damping
T_m	$4 \cdot 10^{3}$ (N/m)	Membrane tension
$T_{\mathrm{s}1}$	138.67 (N)	String 1 tension
$T_{\mathrm{s}2}$	145.53 (N)	String 2 tension
$T_{\mathrm{s}3}$	140.73 (N)	String 3 tension
f_s	$48 \cdot 10^{3}$ (Hz)	Sample rate

Table 2. List of parameter values used when simulating the shamisen. Parameters were taken from * [21]; ** [22]; * [23]; ** [24]; all the lengths† are set to 1 and the other parameters are tuned empirically to produce a desired sound.

q_l^{n+1}, q_l^n and q_l^{n-1} respectively. All of these are initialised to 0.

4.2 String Tuning

The strings are tuned to 'C4', 'G4' and 'C5' as it is a common tuning for the shamisen. The desired pitch for each string is achieved through empirical testing by mainly changing the tension and adjusting the radius of strings before the start of the simulation. Although the other parameters could be adjusted in order to tune the instrument, changing the density of the string or the Young's modulus would not be possible with a real instrument so it was decided to leave these parameters out when tuning.

4.3 Excitation

The system is excited by "plucking" one of the components. Simplified plucking is modeled as a raised cosine added to the current and previous states of the component. 1D components like the string and the bridge use a one dimensional raised cosine, where the membrane uses a 2D version of this as its excitation. A more realistic excitation model is left for future work.

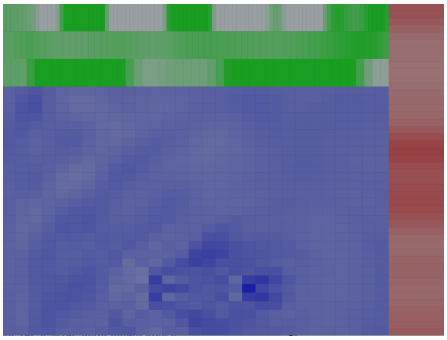


Figure 3. GUI of the digital shamisen. Three green rows at the top are the three strings, red column on the right is the bridge and the blue matrix is the membrane.

4.4 Output

The output of the shamisen is a sum of the components' outputs. The strings and the bridge have a single output location which is defined independently for all the components. The membrane output is simplified to be is a sum of the displacements of all the grid points. Depending on the membrane grid size the gain of the output has to be adjusted according to the number of samples in the scheme.

4.5 Order of Calculations

The pseudocode shown in Algorithm 1 gives a very high-level overview of the order of the calculations done in the application.

```
Initialise the parameters;
Calculate the number of grid points N;
Initialise the state vectors ;
while The application is running do
    if Mouse click on the component then
        Excite the component;
    end
    Calculate the schemes;
    Calculate the connection forces;
    Add the force to the schemes;
    Retrieve output sound;
    Update the states: q_l^{n-1} = q_l^n;  q_l^n = q_l^{n+1};
end
```

Algorithm 1: Pseudocode showing the order of calculations when the program is started and running.

4.6 Graphical User Interface

The graphical user interface (GUI) has a simplistic design, where the state variables q_l^n are displayed as shown in Figure 3. Three different colours were chosen, to indicate the three different types of elements. The intensity

of the colour indicates the displacement of q at location $\boldsymbol{l}$. The graphical representation is more suited for showing the normalized vibrations of the different components rather than visualising the instrument itself. Just clicking on the component will excite it and along with the auditory feedback, a graphical representation of what is happening will be shown. The graphics are updated at a rate of 15 Hz.

5. RESULTS AND DISCUSSION

Informal listening tests by the authors and some fellow students have confirmed that the shamisen has a specific attack sound that when compared with the recreated shamisen exhibited similar timbral qualities. Naturally, formal listening tests need to be conducted to verify this. The output spectrogram of the three strings excited in succession is visualised in Figure 4.

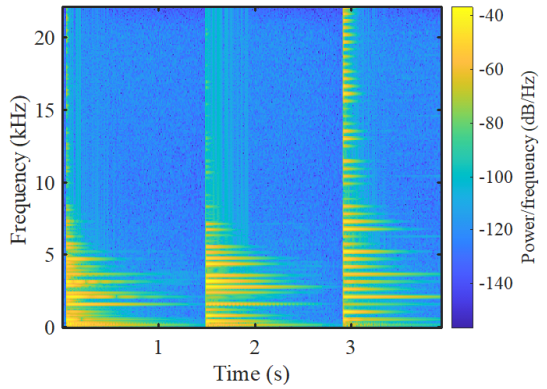


Figure 4. A spectrogram of the three shamisen strings being excited in a succession from the lowest one to the highest. The audio sample used to create this spectrogram is available at [25].

Concerning computational cost, the CPU usage of the real-time application was tested using a MacBook Pro with a 2.2 GHz Intel i7 processor. The strings were continuously excited during the profiling process. The full application uses ca. 51% CPU, whereas the physical model alone without the graphics update uses ca. 39% CPU. This shows that the physical model can easily run in real-time.

As the excitation is adding to the states, sometimes the audio can start clipping, in some cases producing a desirable percussive sound of the membrane and in some cases producing a digital clipping sound, which can affect the timbre of the instrument.

6. CONCLUSION AND FUTURE WORK

In this paper, a real-time implementation of a physical model of the shamisen has been presented. The physical model is based on FDTD methods and was implemented in C++ using the JUCE framework. Informal evaluations show that, while lacking the buzzing, the sound has been found natural but not entirely faithful to the real instrument sound.

Future work includes an addition of the buzzing to the instrument timbre which could possibly be achieved by a FDS collision modeling as described in [26] or a modal collisions model by authors of [27]. Furthermore, a comparison between the different sized membranes is needed in order to find the balance of the audio quality and the CPU usage.

Lastly, a physical implementation of the the "fretting" and "plucking" would benefit the model by adding the expressiveness. It would be nice to model "fretting" as done by the authors of [28] and the "plucking" as described in [29–31]. Alternatively a less costly functional transformation method could be applied to model "fretting" [32].

7. REFERENCES

[1] Brussels Musical Instruments Museum, "Shamisen," 2021. [Online]. Available: http://www.mim.be/shamisen

[2] B. Ono, "How to play a shamisen/way to play a shamisen is explained here in the picture and moving image." 2009. [Online]. Available: https://www.shamisen.info/ehikikatasan/kotonosoho.html

[3] KogeJoe, "File:heike shamisen.jpg - wikimedia commons," 2011. [Online]. Available: https://commons.wikimedia.org/wiki/File:Heike_shamisen.JPG

[4] J. Champion, "Shamisen instrument samples | free wave samples," 2021. [Online]. Available: https://freewavesamples.com/instrument/shamisen

[5] Syntheway Virtual Musical Instruments, 2021. [Online]. Available: https://syntheway.com/ShamiKoto.htm

[6] Impact Soundworks, "Koto nation by impact soundworks (VST, AU, AAX)," 2021. [Online]. Available: https://impactsoundworks.com/product/koto-nation/

[7] Sonica Instruments, "Sanshin - sonica instruments," 2021. [Online]. Available: http://sonica.jp/instruments/index.php/en/products/sanshin

[8] J. O. Smith, "Physical modeling using digital waveguides," *Computer Music Journal*, 1992.

[9] C. Erkut and V. Välimäki, "Model-based sound synthesis of tanbur, a turkish long-necked lute," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2000, pp. 769–772.

[10] F. Germain and G. Evangelista, "Synthesis of guitar by digital waveguides: Modeling the plectrum in the physical interaction of the player with the instrument," in *2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2009, pp. 25–28.

[11] S. Bilbao, *Numerical Sound Synthesis*. John Wiley & Sons, Ltd, 2009.

[12] C. Erkut and M. Karjalainen, "Finite difference method vs. digital waveguide method in string instrument modeling and synthesis," in *Proceedings of the International Symposium on Musical Acoustics (ISMA-02)*, 2002.

[13] ——, "Digital waveguides versus finite difference structures: Equivalence and mixed modeling," *EURASIP Journal on Advances in Signal Processing*, 2004.

[14] M. van Walstijn, J. Bridges, and S. Mehes, "A real-time synthesis oriented tanpura model," in *Proceedings of the 19th International Conference of Digital Audio Effects (DAFx-16)*, 2016.

[15] J. Woodhouse, D. Politzer, and H. Mansour, "Acoustics of the banjo: measurements and sound synthesis," *Acta Acustica*, vol. 5, 2021.

[16] M. Ducceschi and C. Touzé, "Simulations of nonlinear plate dynamics: an accurate and efficient modal algorithm," in *Proceedings of the 18th International Conference on Digital Audio Effects (DAFx-15)*, 2015.

[17] F. Avanzini and R. Marogna, "A modular physically based approach to the sound synthesis of membrane percussion instruments," *IEEE Transactions on Audio, Speech, and Language Processing*, 2010.

[18] F. Pfeiffle and R. Bader, "Real-time physical modelling of a complete banjo geometry using FPGA hardware technology," 2009. [Online]. Available: http://systmuwi.de/Pdf/Papers/Bader%20papers/Physical%20Modeling/PhysicalModeling_Banjo/Pfeifle,Bader_04FPGA_Format.pdf

[19] S. Willemsen, S. Serafin, S. Bilbao, and M. Ducceschi, "Real-time implementation of a physical model of the tromba marina," in *Proceedings of the 17th Sound and Music Computing Conference*, 2020.

[20] JUCE, "JUCE," 2020. [Online]. Available: https://juce.com/

[21] "Bachido home," 2021. [Online]. Available: https://bachido.com/

[22] F. Ko, S. Kawabata, M. Inoue, M. Niwa, S. Fossey, and J. Song, "Engineering properties of spider silk fibers," in *MRS Proceedings*, 2001.

[23] "Engineering toolbox," 2001. [Online]. Available: https://www.engineeringtoolbox.com/

[24] F. Azzarto, "What you need to know about...drumheads | modern drummer magazine," 2010. [Online]. Available: https://www.moderndrummer.com/2011/10/what-you-need-to-know-about-drumheads/

[25] T. Lasickas, "Shamisen-juce: Real-time shamisen," 2021. [Online]. Available: https://github.com/titas2001/Shamisen-JUCE

[26] M. Ducceschi and S. Bilbao, "Non-iterative solvers for nonlinear problems: the case of collisions," in *Proceedings of the 22nd International Conference of Digital Audio Effects (DAFx-19)*, 2019.

[27] C. Issanchou, J.-L. Le Carrou, S. Bilbao, C. Touzé, and O. Doaré, "A modal approach to the numerical simulation of a string vibrating against an obstacle: Applications to sound synthesis," in *Proceedings of the 19th International Conference on Digital Audio Effects (DAFx-16)*, 2016.

[28] S. Bilbao and A. Torin, "Numerical modeling and sound synthesis for articulated string/fretboard interactions," in *Journal of the Audio Engineering Society*, vol. 63, 2015.

[29] S. Willemsen, N. Andersson, S. Serafin, and S. Bilbao, "Real-time control of large-scale modular physical models using the sensel morph," in *Proceedings of the 16th Sound and Music Computing Conference*, 2019.

[30] S. Bilbao, M. Ducceschi, and C. Webb, "Large-scale real-time modular physical modeling sound synthesis," in *Proceedings of the 19th International Conference on Digital Audio Effects (DAFx-16)*, 2019.

[31] P. R. Cook, *Real Sound Synthesis for Interactive Applications*. A K Peters, Ltd, 2002.

[32] L. Trautmann and R. Rabenstein, "Multirate simulations of string vibrations including nonlinear fret-string interactions using the functional transformation method," *EURASIP Journal on Advances in Signal Processing*, 2004.

IMPLEMENTING PHYSICAL MODELS IN REAL-TIME USING PARTITIONED CONVOLUTION: AN ADJUSTABLE SPRING REVERB

Marius G. ONOFREI (monofr11@student.aau.dk) [1], **Silvin WILLEMSEN** (sil@create.aau.dk) [2], and **Stefania SERAFIN** (sts@create.aau.dk) [2]

[1] *Dept. of Architecture, Design & Media Technology*, **Aalborg University Copenhagen**, Denmark
[2] *Multisensory Experience Lab, CREATE*, **Aalborg University Copenhagen**, Denmark

ABSTRACT

This paper presents a detailed description of the development of a real-time spring reverb effect interface which is built based on the solution of a complex physical modelling implementation of helical springs. Impulse responses for various helical springs with different physical parameters are computed offline using an implicit finite difference scheme. These are then used to manipulate a real-time input sound by means of a partitioned convolution implementation, which allows for the use of long impulse responses with low latency. Furthermore, a smooth transition is carried out when changing from one impulse response to another, thus providing the means for real-time manipulation of the physical parameters of the spring and consequently the effect's quality.

1. INTRODUCTION

Spring reverbs have been around since the 1940s [1], and have been developed as cheap, compact devices which give the illusion of room-reverberation [2]. However, due to their highly dispersive nature, they could never really reproduce the sound of a real room. Even so, their sound had its unique appeal and they became very popular, particularly due to their affordability and compact size which allowed them to be included in classic guitar amplifiers throughout the late 20th century. Spring reverb simulations have been implemented for example using combinations of allpass filters [3] or other combinations of filters and delays [4], [5], as well as virtual analogue simulations [6].

In [7], Bilbao and Parker suggest a spring reverb simulation using finite difference schemes (FDSs), based on a helical spring model where the pitch angle is assumed to be small, described as a system of two variables: the transverse and longitudinal vibration. The model is a reduced version of the twelve variable system described by Wittrick [8]. Starting from the same model, Bilbao describes a more complex interleved FDS which includes the effect of the pitch angle in [9]. Van Walstijn presents an alternative FDS which uses ghost nodes in order achieve higher order spatial accuracy in [10], but needs to be evaluated a a very high sample rate of 1 MHz in order to limit numerical dispersion.

The implicit method proposed by [7] can be run at a typical sampling frequency of 44100 Hz and capture the complex dispersive properties of the spring and could potentially be implemented as-is in real-time for a fixed set of spring parameters. This, however, requires a large dimensional matrix inversion for solving the update equation. Additionally if one desires to change the spring's physical parameters dynamically, an optimization procedure is necessary in order to calculate values for the free parameters of the scheme as to minimize the difference between the numerical dispersion and the dispersion of the original system. This computationally demanding fact results in this solution being impractical for a real-time implementation.

The main aim in our paper is to make use of Bilbao and Parker's helical spring physical model in a real-time spring reverb application regardless of the limitation described in the above paragraph. Using this model gives us the flexibility of easily simulating springs of different physical properties. We digitally reproduce a spring reverb and embed it in a physical interface aiming to extend its possibilities compared to the typical uses, with a focus being on the real-time manipulation of the spring's physical parameters. Our approach consists of computing a database of helical spring impulse responses, where the physical parameters of the springs are varied in a consistent way with a focus on the parameters found to be most critical in terms of audio perception. These impulse responses can then be convolved with a dry input sound in order to add the wet spring reverb quality. This is achieved by implementing a partitioned convolution algorithm which allows for the use of long duration impulse responses with minimal latency, as described in [11].

Using this convolution approach one could be tempted to skip the physical model part and use measured impulse responses of mechanical spring reverb units instead. This, however, would reduce the flexibility of changing the spring physical parameters as desired and achieving a database with smoothly varying parameters.

Furthermore, an additional feature is developed in order to achieve a smooth transition of sound when changing between the various impulse responses. This is an addition compared to what the first author found as readily available implementations of partitioned convolution, such as the one available in the SuperCollider platform [12], where

one cannot switch from one impulse response to another without stopping the sound.

2. HELICAL SPRING

2.1 Continuous Model

A simple model for the vibration of a helical structure, which can reasonably simulate the behavior in the human auditory range is given by Bilbao and Parker in [7] which follows their previous work in [13]. They propose a coupled model where the transverse displacement u, and longitudinal displacement, ζ, along the arclength $x \in [0, L]$, for some unwound spring length L, are coupled. This is described in continuous time by the following system of PDEs, where the subscripts t and x denote a derivative with respect to time and space respectively:

$$u_{tt} = \frac{-Er^2}{4\rho}\left(u_{xxxx} + 2\epsilon^2 u_{xx} + \epsilon^4 u\right) + \frac{E\epsilon}{\rho}\left(\zeta_x - \epsilon u\right) - 2\sigma_t u_t, \tag{1a}$$

$$\zeta_{tt} = \frac{E}{\rho}\left(\zeta_{xx} - \epsilon u_x\right) - 2\sigma_l \zeta_t, \tag{1b}$$

where r is the radius of the wire, which is of circular cross-section and the parameter $\epsilon \approx 1/R$ is a measure of the curvature of the spring, with R being the coil radius. Furthermore, parameters E and ρ are related to the material of the spring, the first being the Young's modulus of elasticity and the second being the material density. The parameters σ_l and σ_t model loss in the longitudinal and transverse direction respectively. The number of physical parameters of the spring can be reduced by rewriting the system in Equation (1) in a scaled form. This can be done by introducing the non-dimensional variables $x' = x/L$, $u' = \epsilon u$ and $\zeta' = \zeta/L$. Furthermore, one can write $\kappa = (r\sqrt{E/\rho})/(2L^2)$ as a measure of the stiffness of the spring and $\gamma = \sqrt{E/\rho}/L$ as the longitudinal wave velocity, both measured in s^{-1}. Then the curvature is normalized as the dimensionless parameter $q = \epsilon L$. This results in the following system (the $'$ superscript that indicates the 'scaled' parameter was removed for brevity):

$$u_{tt} = -\kappa^2\left(u_{xxxx} + 2q^2 u_{xx} + q^4 u\right) + q^2\gamma^2\left(\zeta_x - u\right) - 2\sigma_t u_t, \tag{2a}$$

$$\zeta_{tt} = \gamma^2\left(\zeta_{xx} - u_x\right) - 2\sigma_l \zeta_t. \tag{2b}$$

One can investigate the typical dispersive characteristic of the helical spring by deriving the dispersion relationship of the system given in Equation (2) in the lossless case. This can be done by introducing solutions of the form $u(x,t) = Ue^{j(\omega t+\beta x)}$ and $\zeta(x,t) = Ze^{j(\omega t+\beta x)}$, where U and Z are some constants. Solving for the nontrivial solutions of the resulting system of equations the following relationship governing the dispersion is obtained:

$$(\omega^2 - \gamma^2\beta^2)(\omega^2 - \kappa^2(\beta^2 - q^2)^2 - \gamma^2 q^2) - \gamma^4 q^2 \beta^2 = 0. \tag{3}$$

This results in two separate dispersion relationships, i.e. pairs of (ω, β) functions that satisfy Equation (3), one of which lies above the limit of human hearing. The other solution, which lies in the audiable range, is illustrated in Figure 1 for a spring with the parameters as given in the caption, where $f = \omega/2\pi$. This figure is highly illustrative of the the interesting behavior of helical springs. It can be seen that dispersion relationship is not monotonic, meaning that components of different wave lengths can have the same temporal frequency. Furthermore, it can be seen that the dispersion relationship shown has a zero at $\beta = q$ (vertical line in the Figure 1) and has a maximum in the lower frequency range at approximately $\beta = q/2$. This is the wave length which corresponds to the frequency f_c given by:

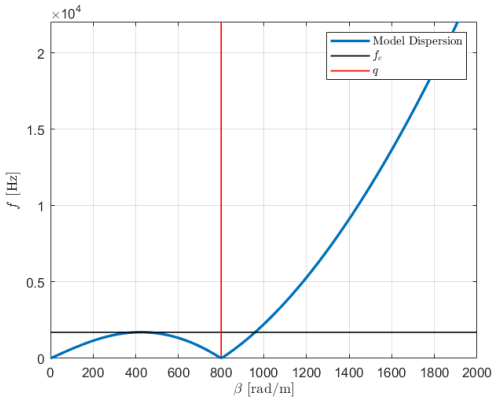


Figure 1. Auditory range solution to the dispersion relationship for a helical spring system, with $\kappa = 0.05$, $\gamma = 2000$ and $q = 800$.

$$f_c = \frac{3\kappa q^2}{8\pi\sqrt{5}}, \tag{4}$$

which is the transition frequency where the dispersion regime changes, and is of perceptual interest for reverberation purposes.

2.2 Finite Difference Scheme

The FDS used for solving this system is an implicit scheme proposed by Bilbao and Parker in [7]. First, the continuous time-space domain over which the PDEs are defined is discretized across a time-space grid of $t = nk$ and $x = lh$, such that the grid function u_l^n denotes a discretized version of $u(x,t)$. The same applies for ζ where $\zeta(x,t) \approx \zeta_l^n$. Then h is the spatial step of the discretization and k is the time step, which results from a desired sampling frequency of the model from $k = 1/f_s$. Furthermore, l and n are integers indexing space and time respectively. As a means of approximating the derivatives in the continuous-time PDEs given in Equation (2), the following finite difference operators are introduced, as per [14]:

$$u_t \approx \delta_{t\cdot} u_l^n = \frac{1}{2k}\left(u_l^{n+1} - u_l^{n-1}\right), \tag{5a}$$

$$u_{tt} \approx \delta_{tt} u_l^n = \frac{1}{k^2}\left(u_l^{n+1} - 2u_l^n + u_l^{n-1}\right), \tag{5b}$$

$$u_x \approx \delta_{x-} u_l^n = \frac{1}{h}\left(u_l^n - u_{l-1}^n\right), \tag{5c}$$

$$u_x \approx \delta_{x+} u_l^n = \frac{1}{h}\left(u_{l+1}^n - u_l^n\right), \tag{5d}$$

$$u_{xx} \approx \delta_{xx} u_l^n = \frac{1}{h^2}\left(u_{l+1}^n - 2u_l^n + u_{l-1}^n\right), \tag{5e}$$

$$u_{xxxx} \approx \delta_{xxxx} u_l^n = \frac{1}{h^4}(u_{l+2}^n - 4u_{l+1}^n + 6u_l^n - 4u_{l-1}^n + u_{l-2}^n), \tag{5f}$$

$$u \approx \mu_{t\cdot} u_l^n = \frac{1}{2}\left(u_l^{n+1} + u_l^{n-1}\right). \tag{5g}$$

With this framework, a discretization of the system of equations given in Equation (2) can be written in the following way [7]:

$$\begin{aligned}(1+\eta\kappa k\delta_{xx})\delta_{tt} u = &-\kappa^2\left(\delta_{xxxx} u + 2\overline{q}^2\delta_{xx} u + \overline{q}^4 u\right)\\ &+ \gamma^2 q^2(\alpha + (1-\alpha)\mu_{t\cdot})\left(\delta_{x-}\zeta - u\right)\\ &- 2\sigma_t \delta_{t\cdot} u,\end{aligned} \tag{6a}$$

$$\begin{aligned}(1+\theta\gamma^2 k^2\delta_{xx})\delta_{tt}\zeta = &\ \gamma^2(\alpha + (1-\alpha)\mu_{t\cdot})\left(\delta_{xx}\zeta - \delta_{x+} u\right)\\ &- 2\sigma_l \delta_{t\cdot}\zeta.\end{aligned} \tag{6b}$$

For explicit numerical schemes one can calculate an update grid function value at a location $x = lh$, i.e. u_l^{n+1}, knowing the values of the grid function u at current (u_l^n) and previous time steps (u_l^{n-1}). Such methods however need to be run at very high sample rates in order to accurately model the dispersion of the system within an audible bandwidth [14]. This can be alleviated by the use of an implicit scheme instead, where the update solution needs to be computed at multiple locations along the grid functions. Expanding the difference operators in Equation (6), it is found that in order to compute the future value at the update point u_l^{n+1}, future values at its neighboring points: u_{l+1}^{n+1} and u_{l-1}^{n+1} are needed, hence a linear coupling among the unknown values of the grid function is introduced. This is due to the introduction of a number of the difference operators δ_{xx} to the left side of the equations and $\mu_{t\cdot}$ to the coupling terms (between u and ζ). The "weights" of these operators are controlled by a number of free parameters in the numerical scheme: η, θ, α. For example if these three parameters are all taken as 0, then the scheme in Equation (6) reduces to an explicit scheme. These parameters can be tuned to provide a more accurate solution in terms of dispersion for a given set of physical spring parameters.

Following the suggestion in [14], $\alpha = 1/2$ and an additional parameter $\overline{q} = (2/h)\sin(qh/2)$ is introduced as an approximation to q. The remaining parameters η and θ are free to change.

Similar to how the dispersion relationship was computed for the continuous model in Equation (3), the numerical dispersion can be derived by inserting the discretized form of the test solutions, i.e. $u_l^n = Ue^{j(\omega kn+\beta lh)}$ and $\zeta_l^n = Ze^{j(\omega kn+\beta lh)}$ in Equation (6). Figure 2 shows a comparison of the numerical dispersion with the model dispersion for different choices of the free parameters. The effect of the choice of $\overline{q}$ is illustrated going from (a) to (b), while the effect of tuning the η and θ parameters is seen going from (b) to (c). The optimal η and θ values differ for each possible combination of spring parameters, hence an optimization procedure is used to compute these optimal values by means of introducing a mean squared error loss function, $\mathcal{L}$ between the values of ω_{model}, resulting from the continuous model dispersion relationship and ω_{FDS}, which results from the dispersion relationship of the FDS, as given in Equation (7):

$$\begin{aligned}\mathcal{L} &= \frac{1}{M}\sum_{i=1}^{M}(\omega_{\text{model},i} - \omega_{\text{FDS},i})^2,\\ \omega_{\text{model},M} &< \pi f_{\text{s}},\end{aligned} \tag{7}$$

where i is an index of the ω values and $M+1$ is the index at which ω_{model} is bigger than the range of interest. This loss is minimized with respect to the η and θ parameters via a Nedler-Mean simplex algorithm as described in [15]. If one would desire to change the physical parameters of the spring in real-time this optimization needs to be recomputed.

Having a fixed time step k, then h needs to be computed such that the numerical solution remains stable. In [7], an energy-based stability analysis is presented which results in the following stability conditions for h:

$$h \geq 2\gamma k\sqrt{\theta^+}, \tag{8a}$$

$$h \geq \sqrt{\kappa k\left(2\eta^+ + \sqrt{4(\eta^+)^2 + (1+|\cos(qh)|)^2}\right)}, \tag{8b}$$

where $\eta^+ = (\eta + |\eta|)/2$ and $\theta^+ = (\theta + |\theta|)/2$, describe the positive parts of η and θ respectively. An h that satisfies the stability conditions as close to equality as possible, will result in a more accurate numerical solution. Once an h value is chosen, the maximum number of grid intervals can be calculated as $N = 1/h$ (since the length of the spring is normalized to 1 in the scaled system).

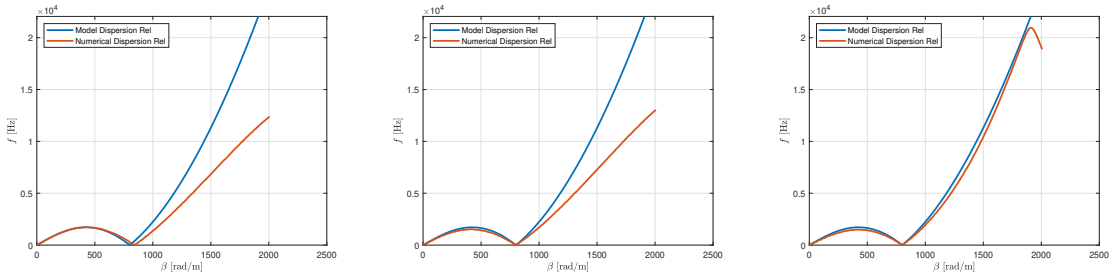
Simply supported boundary conditions are considered for the model at the edges, i.e. $u = u_{xx} = \zeta = \zeta_{xx} = 0$ at $l = 0$ and $l = N$, where N is the number of discretized segments of the scaled helical spring. This means that the domain of calculation will be $l \in [1, 2, ..., N-2, N-1]$, as values at the edges will always be 0.

For the actual implementation of the solver, the finite difference scheme in Equation (6) is rewritten in matrix form. Hence, the finite-length column vectors $\mathbf{u}^n = [u_1^n, ..., u_{N-1}^n]^T$ and $\boldsymbol{\zeta}^n = [\zeta_1^n, ..., \zeta_{N-1}^n]^T$ are introduced over the spatial domain (with $l \in [1, \ldots, N-1]$), where T denotes the transpose. The resulting matrix equations are then factorized with respect to $\mathbf{u}^{n+1}$, $\mathbf{u}^n$, $\mathbf{u}^{n-1}$, $\boldsymbol{\zeta}^{n+1}$, $\boldsymbol{\zeta}^n$ and $\boldsymbol{\zeta}^{n-1}$ resulting in a system of the form

$$\begin{aligned}&\mathbf{A}_1\mathbf{u}^{n+1} + \mathbf{B}_1\mathbf{u}^n + \mathbf{C}_1\mathbf{u}^{n-1}\\ &+ \mathbf{D}_1\boldsymbol{\zeta}^{n+1} + \mathbf{E}_1\boldsymbol{\zeta}^n + \mathbf{F}_1\boldsymbol{\zeta}^{n-1} = 0,\end{aligned} \tag{9a}$$

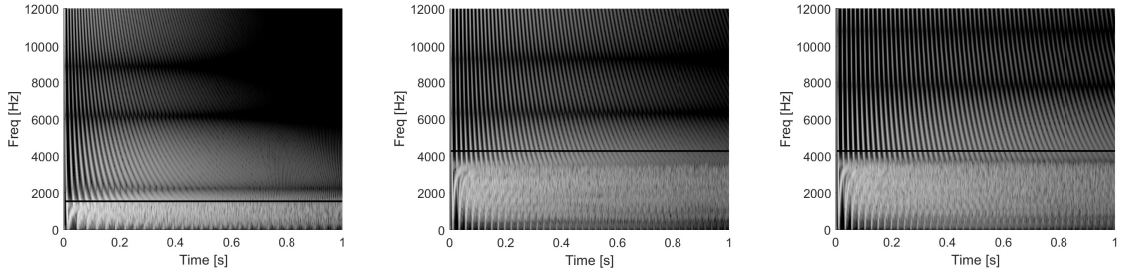
$$\begin{aligned}&\mathbf{A}_2\mathbf{u}^{n+1} + \mathbf{B}_2\mathbf{u}^n + \mathbf{C}_2\mathbf{u}^{n-1}\\ &+ \mathbf{D}_2\boldsymbol{\zeta}^{n+1} + \mathbf{E}_2\boldsymbol{\zeta}^n + \mathbf{F}_2\boldsymbol{\zeta}^{n-1} = 0,\end{aligned} \tag{9b}$$

for Equations (6a) and (6b) respectively. This can be further merged by introducing a state vector which concatenates $\mathbf{u}^n$ and $\boldsymbol{\zeta}^n$ as: $\mathbf{w}^n = [u_1^n, ..., u_{N-1}^n, \zeta_1^n, ..., \zeta_{N-1}^n]^T$.



(a) (b) (c)

Figure 2. Comparisons of model dispersion of a helical spring with physical parameters $\kappa = 0.05$, $\gamma = 2000$ and $q = 800$ with numerical simulations using different sets of free parameters. (a) $\overline{q} = q$, $\eta = \theta = 0$. (b) $\overline{q} = \frac{2}{h}\sin(qh/2)$, $\eta = \theta = 0$. (c) $\overline{q} = \frac{2}{h}\sin(qh/2)$, $\eta = 0.4313$, $\theta = 0.000327$.



(a) (b) (c)

Figure 3. Spectrogram of the impulse response for three helical springs. Black lines represent the model transition frequency f_c. (a) $\kappa = 0.08$, $q = 600$, $\gamma = 1800$. (b) $\kappa = 0.08$, $q = 1000$, $\gamma = 1800$. (c) $\kappa = 0.08$, $q = 1000$, $\gamma = 1000$.

This results in

$$\begin{aligned} &\mathbf{A}\mathbf{w}^{n+1} + \mathbf{B}\mathbf{w}^{n} + \mathbf{C}\mathbf{w}^{n-1} = 0, \quad \text{where,} \\ &\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{D}_1 \\ \mathbf{A}_2 & \mathbf{D}_2 \end{bmatrix} \\ &\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{E}_1 \\ \mathbf{B}_2 & \mathbf{E}_2 \end{bmatrix} \\ &\mathbf{C} = \begin{bmatrix} \mathbf{C}_1 & \mathbf{F}_1 \\ \mathbf{C}_2 & \mathbf{F}_2 \end{bmatrix} \end{aligned} \tag{10}$$

from which an update equation to the state vector can be computed as

$$\mathbf{w}^{n+1} = \mathbf{A}^{-1}\left(-\mathbf{B}\mathbf{w}^{n} - \mathbf{C}\mathbf{w}^{n-1}\right). \tag{11}$$

As can be seen, a large matrix inversion is necessary for solving the system, which along with the optimization of the scheme's free parameters necessary to minimize the numerical dispersion, causes the solution to be impractical with regards to a direct real-time implementation.

3. IMPLEMENTATION

This section details the implementation of the helical spring presented in the previous section and the development of the spring reverb interface. A demonstrative video can be found at [16].

3.1 Impulse Response Database

Impulse responses are generated by exciting the helical spring at one end (at the first free point, i.e. $l = 1$) with a sine sweep covering the human auditory perception range, 20 Hz to 20 kHz and measuring the output at the other end of the spring (the last free point, i.e. $l = N - 1$). Both the excitation and pickup are carried out in terms of the transverse displacement of the spring u, and is consistent with [10]. The initial sine sweep is then deconvolved from this output leaving only the impulse response of the system. For the loss parameters, values suggested in [14] are used: $\sigma_l = \sigma_t = 1.65$.

A database of impulse responses is generated with the aim that there is a distinct difference between each impulse response. This is achieved by systematically changing the parameters of the model in a well-informed manner. Looking again at the dispersion relationship of the continuous model, it is clear that a highly important parameter is q, as it governs both the location where there is a zero, as well as having a big weight in the transition frequency f_c, as it is squared in the numerator in Equation (4). Furthermore, the γ parameter is a scaled version of the longitudinal wave velocity and therefore plays a role in the density of the echoes of the spring.

This can be seen in Figure 3 where two distinct dispersion zones appear: for frequencies above the transition f_c

the behavior is essentially bar-like with higher frequencies travelling faster, while below f_c solutions of differing wave-numbers are possible, [14]. Comparing (a) to (b) one can see that a higher q value raises the transition frequency, while comparing (b) to (c) one can see that a higher γ leads to greater echo density.

For the use in the spring reverb interface, it was decided to keep $\kappa = 0.08$ as a constant value and only vary the other two parameters. Since variations in q are heuristically found to be more distinct perceptually, a denser variation for this parameter is chosen. In total, 27 values are used ranging from 200 to 600 in increments of 25, then from 600 to 1000 in increments of 50 and from 1000 to 1200 in increments of 100. This was done to somewhat mimic the fact that the relationship between frequency and pitch is exponential. This holds when remembering that f_c is directly proportional to q^2. Furthermore two values of γ are considered, 1000 and 1800. These are found to provide a good distinction in the sound of the resulting impulse responses. A smaller difference in these values is not very noticeable, while for significantly higher values of γ, the accuracy of the numerical solution in terms of numerical dispersion suffers. As a result of these choices, this leads to a total of 54 impulse responses. This chosen variation in the spring physical parameters directly translates to a variation in f_c, the dispersion regime transition frequency, by means of Equation (4). Furthermore, a measure of the delay time of the springs in the low frequency dispersive region, T_d, can be derived based on the spring parameters using a relationship given by Parker and Bilbao in [13], $\mathrm{T}_d \approx 4LR/(r\sqrt{E/\rho})$. The average T_{40} decay time of these impulse responses is calculated to be of 2.12 seconds, using Schroeder's backward integration method [17].

3.2 Impulse Responses - Convolution

Since the helical spring system is linear time invariant (LTI) it follows that if one knows the impulse response of this system one can determine its output via the theory of convolution [18]. In time domain this is expressed as

$$y_c[n] = x_c[n] * h_c[n] = \sum_{m=-\infty}^{+\infty} (x_c[m]h_c[n-m]), \quad (12)$$

where x_c is the input to a system which is described by the impulse response h_c and y_c is its output. The subscript c (for convolution) is used to avoid confusion with the variables used in the previous section.

However, convolution is better implemented in frequency domain where Equation (12) transforms into simple multiplication, after which a conversion back to time domain can be performed:

$$\begin{aligned} Y(\omega) &= X(\omega)H(\omega), \\ y_c[n] &= \mathrm{IDFT}(\mathrm{DFT}(x_c[n])\mathrm{DFT}(h_c[n])), \end{aligned} \quad (13)$$

with X being the frequency response of the input, H the frequency response of the impulse response and finally Y is the frequency response of the output, while DFT and IDFT are abbreviations for the discrete Fourier transform and its inverse.

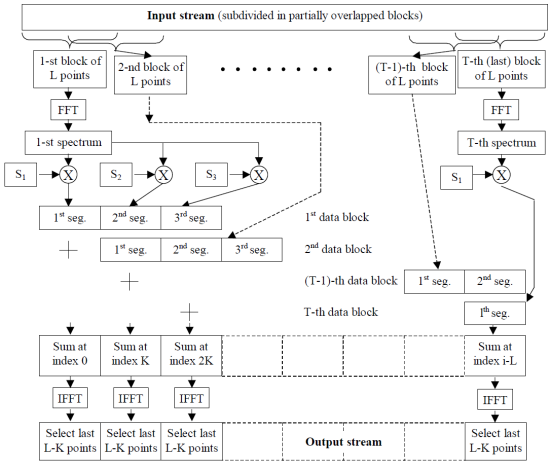


Figure 4. Partitioned convolution process overview, from [11].

Using this method, the length of the impulse response will immediately translate to the latency of the system (excluding other overheads). This is a straightforward and efficient method when the impulse response is of a small length, but the impulse responses of the springs are 2 seconds, or 88200 samples long. A better implementation of convolution for the current task is partitioned convolution, described in detail in [11]. It consists of partitioning the impulse response, h_c, into equally sized blocks of length K. Furthermore, these blocks are treated as separate impulse responses and are convolved using a standard overlap-and-save method [11]. That is, each block is zero-padded to a size $L = 2K$, after which it is transformed to frequency domain via FFT, obtaining a collection of frequency domain filters S. These are then applied to chunks of size L of the input signal while overlapping the results of the latest P input blocks. A diagram of the algorithm is shown in Figure 4, taken from [11]. This is what is used for the current project and the details of the implementation are given in the next section.

A major advantage of the partitioned convolution method is the reduced latency which is only of L rather than 88200 samples.

3.3 Interface Development

The concept behind the interface design is to keep a standard box like design, familiar to most users, while the design of the parameter controls of the interface went hand in hand with the implementation of the algorithm and the various I/O controls. Figure 5 shows the final result.

In order to have the necessary computational power for the heavy partitioned convolution algorithm, as well as to have a standalone interface, it was decided to adopt the Bela platform. The Bela is a small, single-board Linux computer built on the BeagleBone Black that provides high quality audio at ultra-low latency, in addition with a large array of analog and digital I/O options [19]. For this project, the Bela Mini was chosen due to its reduced size

Figure 5. Spring reverb interface

and the fact that it provides all the needed features. Also from the Bela platform, the Bela Trill Square sensor was used, which is a capacitive touch sensor that supports I^2C communication. Details regarding the choice of the sensors are given in Section 3.4.

A standard ABS plastic enclosure was used for embedding the Bela and the various sensor/controls. The electrical connections between the various sensors and the Bela board were done on a standard prototype breadboard using jumper wire cables.

The physical model simulations of the spring impulse responses were carried out in Matlab [20], while the implementation of the partitioned convolution algorithm is carried out in the Bela in-browser IDE in C++. The audio sampling rate used is 44100 Hz, while the analog I/O sample rate is 22050 Hz.

The block size for the audio render is taken as 256 samples, while the block size used in the partitioned convolution algorithm (size L described in Section 3.2) is 8192 samples. This means that the convolution processing function is only run every time 8192 new audio samples are added in the audio input circular buffer. Furthermore, to avoid possible underruns, a multi-threaded implementation is carried out for the convolution processing function, making sure that the audio render function always has higher priority.

Moreover it was found that an initial lag between the global input buffer and the global output buffer of 3 times the hop size K is needed. This basically gives the I/O latency of the interface, which results in 279 ms. Figure 6 shows this latency as measured with the in-browser oscilloscope available in the Bela IDE. If one would reduce the block size L to half, i.e 4096 samples for instance, the resulting latency would also be halved.

However, it was found that the real-time capability of the implementation is limited by the maximum number of frequency domain filter blocks S (see Section 3.2), that is how many overlapped bins the impulse response is divided in. It was found that the program cannot run in real time when more than 22 blocks are used, regardless of the block size

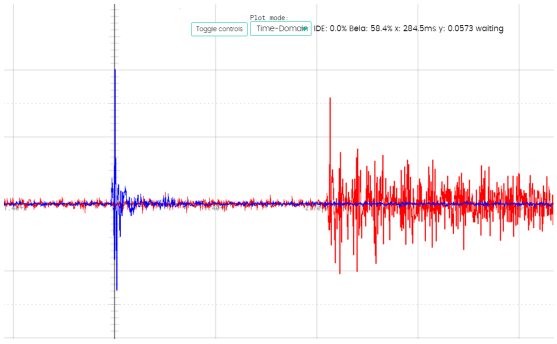


Figure 6. Illustration of system latency as measured with the in-browser oscilloscope available in the Bela IDE. Blue line is the input signal and red line is the output. There is a lag of approximately 279 ms between the two.

L. So when reducing the size from 8192 to 4096 samples, one can only use impulse responses of length $2048 \cdot 23 = 47104$ samples (22 block sizes of length 4096 overlapped with a hop size of 2048), which is a bit more than 1 second at the audio sample rate used. There appears to be a trade-off between the latency of the implementation and the maximum length of the impulse responses.

3.4 Parameter Mapping

Since spring impulse responses from the available database differed with respect to two physical parameters: curvature parameter q and wave velocity γ, it was desired to have the option to quickly navigate through them with respect to both these dimensions. Hence, the Trill Square capacitive sensor from the Bela platform was used, and each impulse response was mapped to locations on the (x,y) grid position on the square. More specifically, the wave speed γ is mapped to the x-axis while curvature parameter q is mapped to the y-axis.

An important feature of the implementation is that a smooth transition from using one impulse response to another is achieved. This was done by doing all the processing related to all the impulse responses in the setup part of the audio algorithm. Then, when moving from one impulse response to another, two convolutions are in fact carried out for a brief transition time chosen as one audio processing bin, i.e., 8192 samples (size L in the partitioned convolution algorithm). A square-root fade-in/fade-out is then carried out between the two resulting outputs. Moreover, when taking the finger off of the sensor the impulse response used for convolution will be the one associated with the last read position.

Other parameter controls are carried out by means of three potentiometers. One is mapped to a global master volume, lineally mapped between -40 dB and +6 dB. Another controls the dry/wet mixture of the output signal, where the minimum value of the potentiometer gives a fully dry sound and the maximum value gives a fully wet sound. The last parameter mapped to the remaining potentiometer is the number of bins of the impulse response considered in the convolution algorithm.

3.5 From Effect to Instrument

Another interesting addition to the interface is the fact that the user can change between two different audio input sources with the use of a toggle switch. The first is an audio-in jack, which can be connected to any other sound producing device, in essence turning the interface into an effect processor. Furthermore, an electret microphone was embedded in a side panel of the box, for which a signal amplifier circuit is built on the breadboard. When switching to the mic input, the interface can be used similarly to an instrument. One can tap, scratch, whistle and speak into the microphone while changing the various real-time parameter controls and produce interesting sounds. Additional investigations regarding this potential use coupled with feedback are planned.

4. CONCLUSION

The work described in this paper focused on combining physical modelling sound synthesis techniques with convolution with the aim to supplement each other towards a real-time implementation of a spring reverb, whose physical parameters can be adjusted on the fly. While the physical model with adjustable parameters proved to be too computationally expensive for a real-time implementation, using partitioned convolution provided a way around this. An important addition to this approach is providing a way to have a smooth transition from the use of one impulse response to another, and the fact that the impulse responses are physically related to each other.

A physical interface was built which allows for an expressive manipulation of an input sound, via the provided parameter controls.

Future work will focus on a more streamlined design of the interface, by means of a CAD software together with laser cutting or 3-D printing manufacturing options. As for the electrical components, a more sturdy strip board implementation followed by a custom PCB design of the circuit is planned. Additional investigations regarding the software implementation can focus on pushing the limits of the model. For instance, how many impulse responses can be easily processed in the audio setup and mapped to the Trill square. Since the physical model for the helical spring is mainly dependent on 3 parameters, an additional dimension representing the κ parameter can be added to the mapping, perhaps using a pressure sensor or making use of the area of the touch.

While the interface was designed with the concept of a spring reverb unit in mind, what it is essentially is a real-time convolution effect processor, with an implemented use case as a spring reverb. One can easily use other impulse responses and completely change the nature of the interface. Such other use cases will be looked into. Lastly, an important observation is that this is a mono audio device, and expanding it to stereo, perhaps mapping different impulse responses to each output channel, would be a welcome feature.

5. REFERENCES

[1] L. Hammond, *"Electrical musical instrument," US Patent No. 2230836*, Feb 1941.

[2] V. Välimäki, J. D. Parker, L. Savioja, J. O. Smith, and J. S. Abel, "Fifty years of artificial reverberation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1421–1448, 2012.

[3] J. S. Abel, D. P. Berners, S. Costello, and J. O. Smith, "Spring reverb emulation using dispersive allpass filters in a waveguide structure," in *Audio Engineering Society Convention 121*. Audio Engineering Society, 2006.

[4] J. S. Abel and E. K. Canfield-Dafilou, "Dispersive delay and comb filters using a modal structure," *IEEE Signal Processing Letters*, vol. 26, no. 12, pp. 1748–1752, 2019.

[5] V. Välimäki, J. Parker, and J. S. Abel, "Parametric spring reverberation effect," *Journal of the Audio Engineering Society*, vol. 58, no. 7/8, pp. 547–562, 2010.

[6] V. Välimäki, S. Bilbao, J. O. Smith, J. S. Abel, J. Pakarinen, and D. Berners, "Virtual analog effects," in *DAFX: Digital Audio Effects*, 2011, pp. 473–522.

[7] S. Bilbao and J. Parker, "A virtual model of spring reverberation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 799–808, 2010.

[8] W. Wittrick, "On elastic wave propagation in helical springs," *International Journal of Mechanical Sciences*, vol. 8, pp. 25–47, 1966.

[9] S. Bilbao, "Numerical simulation of spring reverberation," in *DAFX: Digital Audio Effects*, 2013.

[10] M. Van Walstijn, "Numerical calculation of modal spring reverb parameters," in *Proceedings of the 23rd International Conference on Digital Audio Effects*, 2020, pp. 38–45.

[11] E. Armelloni, C. Giottoli, and A. Farina, "Implementation of real-time partitioned convolution on a dsp board," *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684)*, pp. 71–74, 2003.

[12] S. Wilson, D. Cottle, and N. Collins, *The SuperCollider Book*. The MIT Press, 2011.

[13] J. Parker and S. Bilbao, "Spring reverberation: A physical perspective," 2009.

[14] S. Bilbao, *Numerical Sound Synthesis*. John Wiley and Sons, Ltd, 2009.

[15] J. Lagarias, J. Reeds, M. Wright, and P. Wright, "Convergence properties of the nelder–mead simplex method in low dimensions," *SIAM Journal on Optimization*, vol. 9, pp. 112–147, 1998.

[16] M. G. Onofrei. A physical modelling based adjustable spring reverb effect. Youtube. [Online]. Available: https://www.youtube.com/watch?v=HZRWU0b45vg

[17] M. R. Schroeder, "New method of measuring reverberation time," *The Journal of the Acoustical Society of America*, vol. 37, no. 6, pp. 1187–1188, 1965.

[18] T. H. Park, *Introduction To Digital Signal Processing: Computer Musically Speaking*. World Scientific Publishing Co Pte Ltd, 2008.

[19] A. McPherson and V. Zappi, "An environment for submillisecond-latency audio and sensor processing on beaglebone black," vol. 2, pp. 965–971, 2015.

[20] MATLAB, *version (R2020b)*. Natick, Massachusetts: The MathWorks Inc., 2019.

ON THE TRANSFER FUNCTION OF THE PIECEWISE-CYLINDRICAL VOCAL TRACT MODEL

Tamara SMYTH (trsmyth@ucsd.edu) [1] and **Devansh ZURALE**[1]

[1]**University of California San Diego**, La Jolla, USA

ABSTRACT

In this work, a matrix formulation of a piecewise one-dimensional waveguide model of the vocal tract (having varying cross-sectional area along its length) is used to derive model transfer functions suitable for both cylindrical or conical sections, with outputs tapped at the position of the glottis and the lips. The transfer function tapped at the lips is then considered in more detail for cylindrical waveguide sections and presented in its more useful form as a ratio of polynomial functions in the discrete frequency variable z, with coefficients vectors calculated for two cases: one where model boundaries are scalar losses and the other where losses are dependent on frequency. Through a transfer function with coefficients that are dependent on parameters of cross-sectional area and boundary conditions, the model may not only be controlled in real time, but the relationship to other vocal tract representations, in particular linear prediction coding (LPC) of speech, can be more easily shown, laying the foundation for inverse problems such as parameter estimation and source-filter separation. Finally, a comparison between model transfer function coefficients and those estimated by LPC (which assumes an all-pole filter) is discussed, suggesting that lower-order (and less computationally costly) LPC estimators might benefit from acoustically-informed boundary losses and the resulting introduction of zeros into the transfer function.

1. INTRODUCTION

The work herein borrows strategies for waveguide modeling of wind instrument bores and bells which, like the vocal tract, have shapes that are frequently not cylindrical or conical and thus have no known analytic solution. Though round-trip propagation delay in purely cylindrical and/or conical tubes may be modeled as a single one-dimensional waveguide (bi-directional delayline) element, the vocal tract has a varying cross-sectional area along its length and is better modeled using a piecewise approach (see Figure 1). Here, the vocal tract model is presented from the perspective of musical instrument modeling the desire to have parameters that can be both estimated and controlled in real time. To that end, the theory of piecewise waveguide modeling is reviewed [1–3] and a matrix formulation is presented that leads to parametric transfer functions modeling the vocal tract, one with output tapped at the position of the glottis and the other at the lips, initially with no assumption on whether waveguide sections are cylindrical of conical.

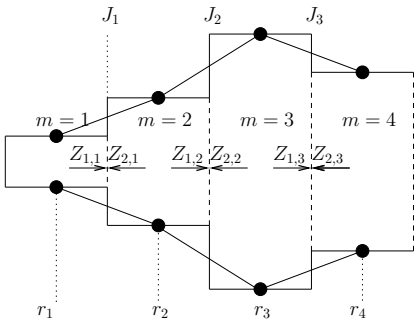


Figure 1. A sequence of $M = 4$ conical/cylindrical sections with radii r_m (and corresponding cross-sectional area) interleaved with $N = M - 1$ scattering junctions.

Though representations of the vocal tract have taken on different forms in the literature several can be made fundamentally equivalent—this is true of LPC [4], Kelly-Lockbaum [5] and piecewise models where sections are made uniformly cylindrical. These techniques model formants in the produced sound as a result of characteristic changes in the vocal tract shape and, under certain basic conditions/configurations, similarly result in an all-pole filter. The estimation of filter feedback coefficients using LPC is a frequently used technique and methods have been proposed to enhance its all-pole approximation by estimating, often iteratively, more accurate losses in the system [6]. Here, the contribution of vocal tract boundaries (lip reflection/transmission) and whether they are modeled as scalar or frequency-dependent losses, is shown to cause a divergence in model similarities. The suggestion is, therefore, that acoustically-informed boundaries as used in the piecewise cylindrical model, and the introduction of zeros into the transfer functions as a result of their inclusion, may enhance the all-pole LPC estimation without requiring higher and more computationally costly filter orders.

In the following, Section 2 reviews the theory of scattering junctions and first derives a scattering matrix for a single junction, then the "chain" scattering matrix for the complete vocal tract model. The matrix representations are then used to derive the model transfer functions in Section 3, applicable to both cylindrical and conical

waveguide sections. Section 4 considers the special case of cylindrical sections, showing a transfer function coefficient vector that is a function of the boundaries, first with an assumption of scalar losses that is most strongly related to LPC, then frequency-dependent losses, represented as a convolution in matrix form, that are more physically informed. Finally, in Section 5 a discussion is made on the relationship between waveguide model and LPC followed by a (preliminary) comparison of how both estimate the known glottal pulse from the output of a physical model.

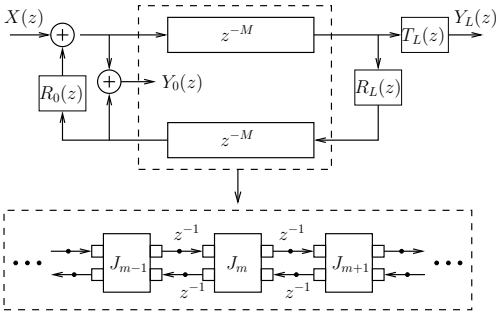


Figure 2. The vocal tract's varying cross-sectional area along its length may be implemented as a piecewise model—a cascade of two-port scattering junctions with interleaved unit-sample bi-directional delays (cylindrical/conical sections), with terminating boundary conditions $R_0(z)$ at the glottis and $R_L(z)$ at the lips and with outputs $Y_0(z)$ and $Y_L(z)$ in response to input $(X(z)$. Wall losses are omitted.

2. VOCAL TRACT SCATTERING MATRIX

As shown in Figures 1 and 2, the one-dimensional piecewise conical/cylindrical model of the vocal tract is comprised of M sections, each a bidirectional unit-sample delay, corresponding to acoustic propagation distance in one time sample, interleaved with $N = M - 1$ two-port $N_p = 2$ scattering junctions.

Scattering, the reflection and transmission of a wave that occurs when there is a change in the wave's characteristic impedance, may be modeled using a multi-port scattering junction where the number of ports N_p is twice the dimensionality of the wave propagation and the wave impedance on each port is determined by the medium (or geometry) it serves to connect. For waves propagating along the length of a *diverging* conical section terminated at port n a distance l_n from the cone apex, the wave impedance is a complex function of frequency given by

$$Z_n(l,\omega) = \frac{\rho c}{S_n} \cdot \frac{j\omega}{j\omega + c/l_n}, \tag{1}$$

where S_n is the cross-sectional at the port, ρ is the medium density and c is the propagation velocity. If the wave is propagating in the opposite direction toward the cone apex, effectively seeing a *converging* conical section, its impedance is given by the complex conjugate,

$$Z_n^*(l,\omega) = \frac{\rho c}{S_n} \cdot \frac{j\omega}{j\omega - c/l_n}. \tag{2}$$

For plane waves traveling in cylindrical sections, the distance l_n to the cone apex is infinite and the characteristic impedance reduces to a real value:

$$Z_n = \rho c / S_n. \tag{3}$$

Each of the junction's ports has a physical pressure p_n and volume velocity U_n that is the sum of wave components propagating in "i" and out "o" of port:

$$p_n = p_n^i + p_n^o, \quad \text{and} \quad U_n = U_n^i + U_n^o. \tag{4}$$

and which are related by the characteristic impedance:

$$U_n^i = \frac{p_n^i}{Z_n}, \quad U_n^o = -\frac{p_n^o}{Z_n^*}, \tag{5}$$

(the negative output volume velocity accounts for the fact that it is a directional quantity and moves in the direction in which it generates pressure [7]). Because the junction is shared by all mediums it connects, *the law for conservation of mass and momentum* dictate that the pressure at the junction be continuous and equal to the pressure on each port:

$$p_J = p_n = p_n^i + p_n^o, \tag{6}$$

and the sum of volume velocity on each port is equal zero,

$$\sum_{n=1}^{N_p} U_n = \sum_{n=1}^{N_p} (U_n^i + U_n^o) = 0. \tag{7}$$

For the two-port ($N_p = 2$) junction used in the piecewise vocal tract model, it follows from (6) that

$$p_1^i + p_1^o = p_2^i + p_2^o, \tag{8}$$

and from (7), with the substitution given by (5), that

$$\frac{p_1^i}{Z_1} - \frac{p_1^o}{Z_1^*} = -\left(\frac{p_2^i}{Z_2} - \frac{p_2^o}{Z_2^*}\right). \tag{9}$$

Equations (8) and (9) may be conveniently expressed in matrix form,

$$\mathbf{C} \begin{bmatrix} p_1^i \\ p_1^o \end{bmatrix} = \mathbf{D} \begin{bmatrix} p_2^i \\ p_2^o \end{bmatrix}, \tag{10}$$

where $\mathbf{C}$ and $\mathbf{D}$ are 2×2 matrices given by

$$\mathbf{C} = \begin{bmatrix} 1 & 1 \\ \frac{1}{Z_1} & -\frac{1}{Z_1^*} \end{bmatrix} \quad \text{and} \quad \mathbf{D} = \begin{bmatrix} 1 & 1 \\ -\frac{1}{Z_2} & \frac{1}{Z_2^*} \end{bmatrix}, \tag{11}$$

and rearranged to yield the expression relating left and right port, $n = 1$ and 2, respectively, input and output pressure wave components:

$$\begin{bmatrix} p_1^i \\ p_1^o \end{bmatrix} = \mathbf{C}^{-1}\mathbf{D} \begin{bmatrix} p_2^i \\ p_2^o \end{bmatrix}, \tag{12}$$

where

$$\mathbf{C}^{-1}\mathbf{D} = \frac{1}{\frac{1}{Z_1^*}+\frac{1}{Z_1}}\begin{bmatrix}\frac{1}{Z_1^*} & 1\\ \frac{1}{Z_1} & -1\end{bmatrix}\begin{bmatrix}1 & 1\\ -\frac{1}{Z_2} & \frac{1}{Z_2^*}\end{bmatrix} = \begin{bmatrix}\frac{Z_1(Z_2-Z_1^*)}{Z_2(Z_1+Z_1^*)} & \frac{Z_1(Z_2^*+Z_1^*)}{Z_2^*(Z_1+Z_1^*)}\\ \frac{Z_1^*(Z_2+Z_1)}{Z_2(Z_1+Z_1^*)} & \frac{Z_1^*(Z_2^*-Z_1)}{Z_2^*(Z_1+Z_1^*)}\end{bmatrix}. \quad (13)$$

As may be seen in Figure 3, the *left* port's input and output wave components are equal to the right and left traveling pressure waves, denoted by $+$ and $-$ superscripts respectively, in section m,

$$\begin{bmatrix}p_1^i\\ p_1^o\end{bmatrix} = \begin{bmatrix}p_m^+\\ p_m^-\end{bmatrix} = \mathbf{p}_m, \quad (14)$$

but the inverse relationship exists between wave components on the junction's *right* port and traveling waves in neighbouring section $m+1$,

$$\begin{bmatrix}p_2^i\\ p_2^o\end{bmatrix} = \begin{bmatrix}p_{m+1}^- z^{-1}\\ p_{m+1}^+ z\end{bmatrix} = \begin{bmatrix}0 & z^{-1}\\ z & 0\end{bmatrix}\mathbf{p}_{m+1}, \quad (15)$$

with vector element ordering made consistent with (14) by multiplying with an antidiagonal matrix that also accounts for the unit-sample delay/advance in one section.

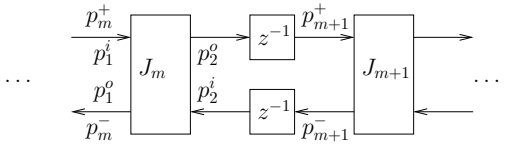


Figure 3. Relationship between the m^{th} junction's left and right (subscript 1 and 2, respectively) port input and output (superscript i and o, respectively) pressure wave components to the right ($+$ superscript) and left ($-$ superscript) traveling pressure waves in adjacent sections m and $m+1$.

With the change to traveling wave variables made by substituting (14) and (15) into (12), the relationship between right and left traveling pressure waves in adjacent sections m and $m+1$ may be given by

$$\mathbf{p}_m = \mathbf{A}_m\mathbf{p}_{m+1}, \quad (16)$$

where the scattering matrix for a single two-port junction

$$\mathbf{A}_m = \left(\mathbf{C}^{-1}\mathbf{D}\right)_m\begin{bmatrix}0 & z^{-1}\\ z & 0\end{bmatrix}, \quad (17)$$

is defined as a product of (13) for the m^{th} junction.

The "chain" scattering matrix for the complete vocal tract model is obtained by first expanding (16),

$$\mathbf{p}_m = \mathbf{A}_m\overbrace{\mathbf{p}_{m+1}}^{\mathbf{A}_{m+1}\mathbf{p}_{m+2}} = \mathbf{A}_m\mathbf{A}_{m+1}\overbrace{\mathbf{p}_{m+2}}^{\mathbf{A}_{m+2}\mathbf{p}_{m+3}} \quad (18)$$

so that traveling pressure waves in the first section can be expressed as a product of those in the final section,

$$\mathbf{p}_1 = \mathbf{P}_{M-1}\mathbf{p}_M, \quad (19)$$

where, for a sequence of M sections and $N = M-1$ junctions, the model's final 2×2 chain scattering matrix is given by the repeated product

$$\mathbf{P}_{M-1} = \prod_{m=1}^{M-1}\mathbf{A}_m = \begin{bmatrix}P_{1,1} & P_{1,2}\\ P_{2,1} & P_{2,2}\end{bmatrix}. \quad (20)$$

3. MODEL TRANSFER FUNCTIONS

To adequately represent the vocal tract so that it may be coupled to a dynamic model of the vocal folds as in [8], it is necessary to obtain two (2) transfer functions representing the model: one with the output pressure tapped at the lips $Y_L(z)$ and the other with the output pressure tapped at the glottis $Y_0(z)$ (see Figure 4).

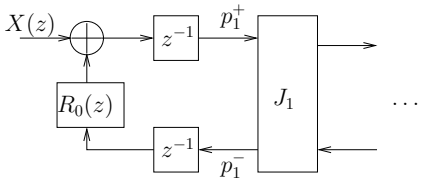


Figure 4. A signal flow diagram of the first waveguide section, showing how input $X(z)$ may be represented as a function of the glottis boundary $R_0(z)$ and traveling pressure waves p_1^+ and p_1^- as given by (21).

Both transfer functions are in response to input pressure $X(z)$ (corresponding to the product of the glottal flow and the characteristic impedance at the entry to the vocal tract) which, following Figure 4, can be defined in terms of the right and left traveling waves in the first section:

$$X(z) = p_1^+(z)z - R_0(z)p_1^-(z)z^{-1}, \quad (21)$$

which, by employing (19) and (20), may then be expressed in terms of traveling waves in the final section

$$X(z) = \left(P_{1,1}p_M^+(z) + P_{1,2}p_M^-(z)\right)z - R_0\left(P_{2,1}p_M^+ + P_{2,2}p_M^-\right)z^{-1}. \quad (22)$$

Finally, using the definition of the open-end lip reflection transfer function $R_L(z) = p_M^-(z)/p_M^+(z)$ and making the substitution $p_M^-(z) = R_L(z)p_M^+(z)$ in (22), the input may be expressed as a function of only the right traveling wave in the final section:

$$X(z) = p_M^+(z)\left(P_{1,1} + P_{1,2}R_L(z)\right)z - p_M^+(z)R_0(z)\left(P_{2,1} + P_{2,2}R_L(z)\right)z^{-1}. \quad (23)$$

The vocal tract transfer function $H_L(z) = Y_L(z)/X(z)$ tapped at the lips is defined as the ratio of output pressure $Y_L(z) = p_M^+(z)T_L(z)$ to input pressure $X(z)$ which, by substituting (23), yields

$$H_L(z) = \frac{T_L(z)z^{-1}}{P_{1,1}+P_{1,2}R_L(z) - R_0\left(P_{2,1}+P_{2,2}R_L(z)\right)z^{-2}}. \quad (24)$$

The transfer function $H_0(z) = Y_0(z)/X(z)$ is the ratio of the pressure at the glottis (vocal tract base)

$$\begin{aligned} Y_0(z) &= X(z) + p_1^-(z)(1 + R_0(z))z^{-1} \\ &= X(z) + \\ &\quad p_M^+(z)(P_{2,1} + P_{2,2}R_L(z))(1 + R_0(z))z^{-1}, \end{aligned}$$

to the system input $X(z)$ given by (23), yielding

$$H_0(z) = \frac{P_{1,1} + P_{1,2}R_L(z) + (P_{2,1} + P_{2,2}R_L(z))z^{-2}}{P_{1,1}+P_{1,2}R_L(z)-R_0\,(P_{2,1}+P_{2,2}R_L(z))\,z^{-2}}, \tag{25}$$

showing how boundary conditions $R_0(z)$ and $R_L(z)$ (further discussed in Section 4.2) and, for a cylindrical section, the assumed amplitude complementary transmission, which for pressure is given by

$$T_L(z) = 1 + R_L(z), \tag{26}$$

contribute to the vocal tract transfer functions.

Though the above transfer functions are sufficient for a frequency-domain representation/implementation of the model, it is preferable to represent it in its more useful form as a ratio of polynomials in the (discrete) frequency variable z, both to allow for time-domain implementation using the corresponding difference equation (obtained by taking the inverse z-transform) and also for comparison (or mapping) to all-pole filter coefficients estimated by LPC.

4. POLYNOMIAL TRANSFER FUNCTION FOR CYLINDRICAL SECTIONS

Though vocal tract sections may be modeled as being either cylindrical or conical (see Figure 1) and the above derivation makes no assumption of either, the section shape is dependent on the choice of expression for impedance (1)-(3) in the matrix given by (13). Conical sections in a time-domain synthesis would require fitting a digital filter to the complex impedances given by (1) and (2) as was done in [3] using the impulse-invariant method [9] and also in [10] using the bilinear transform. Using cylindrical sections, on the other hand, has considerable computational convenience for computing the transfer functions as a ratio of polynomials, as well as allowing better comparison with LPC and related Kelly-Lochbaum models.

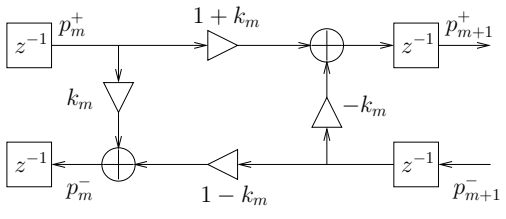


Figure 5. Kelly-Lochbaum scattering junction.

For cylindrical sections, as mentioned in Section 2, the characteristic wave impedance (3) is not a function of frequency but rather a real value inversely proportional to the plane wave's surface area. For waves on left and right ports of the junction between sections m and $m + 1$, this area is the section's cross-sectional area S_m and S_{m+1}, respectively, and the scattering matrix (13) may be reduced to

$$\begin{aligned} \left(\mathbf{C^{-1}D}\right)_m &= \frac{1}{2S_m}\begin{bmatrix} S_m - S_{m+1} & S_m + S_{m+1} \\ S_m + S_{m+1} & S_m - S_{m+1} \end{bmatrix} \\ &= \frac{1}{1 + k_m}\begin{bmatrix} k_m & 1 \\ 1 & k_m \end{bmatrix}, \end{aligned} \tag{27}$$

where the reflection coefficient between sections

$$k_m = \frac{S_m - S_{m+1}}{S_m + S_{m+1}}, \tag{28}$$

is that used in LPC and forms the Kelly-Lochbaum scattering junction shown in Figure 5.

Substituting (27) into (17) yields the single junction scattering matrix between sections sections m and $m + 1$ for the piecewise *cylindrical* model

$$\mathbf{A}_m = \frac{1}{1 + k_m}\begin{bmatrix} k_m & 1 \\ 1 & k_m \end{bmatrix}\begin{bmatrix} 0 & z^{-1} \\ z & 0 \end{bmatrix}, \tag{29}$$

which, for $M = 2$ sections and $N = 1$ junction, yields a model scattering matrix (20) given by

$$\mathbf{P}_1 = \mathbf{A}_1 = \frac{z}{1 + k_1}\begin{bmatrix} 1 & k_1 z^{-2} \\ k_1 & z^{-2} \end{bmatrix}, \tag{30}$$

and for $M = 3$ sections and $N = 2$ junctions,

$$\begin{aligned} \mathbf{P}_2 = \quad & \mathbf{A}_1\mathbf{A}_2 = \mathbf{P}_1\mathbf{A}_2 \\ &= \frac{z}{1 + k_1}\begin{bmatrix} 1 & k_1 z^{-2} \\ k_1 & z^{-2} \end{bmatrix}\frac{z}{1 + k_2}\begin{bmatrix} 1 & k_2 z^{-2} \\ k_2 & z^{-2} \end{bmatrix} \\ &= \frac{z^2}{\prod_{m=1}^{2}(1 + k_m)}\begin{bmatrix} c_0 + c_2 z^{-2} & d_2 z^{-2} + d_0 z^{-4} \\ d_0 + d_2 z^{-2} & c_2 z^{-2} + c_0 z^{-4} \end{bmatrix}, \end{aligned}$$

where polynomial matrix elements have coefficients

$$c_0 = 1, \quad c_2 = k_1 k_2, \quad d_0 = k_1 \quad \text{and} \quad d_2 = k_2. \tag{31}$$

In general, for models having M sections and $N = M - 1$ junctions, the chain scattering matrix is given by

$$\mathbf{P}_N = \prod_{m=1}^{N}\mathbf{A}_m = \mathbf{P}_{N-1}\mathbf{A}_N = \frac{z^N}{\prod_{m=1}^{N}(1 + k_m)}\mathbf{K}_N, \tag{32}$$

where

$$\begin{aligned} \mathbf{K}_N &= \begin{bmatrix} K_{1,1} & K_{1,2} \\ K_{2,1} & K_{2,2} \end{bmatrix} = \prod_{m=1}^{N}\begin{bmatrix} 1 & k_m z^{-2} \\ k_m & z^{-2} \end{bmatrix} \\ &= \begin{bmatrix} \sum_{m=0}^{N-1} c_{2m} z^{-2m} & \sum_{m=1}^{N} d_{2(N-m)} z^{-2m} \\ \sum_{m=0}^{N-1} d_{2m} z^{-2m} & \sum_{m=1}^{N} c_{2(N-m)} z^{-2m} \end{bmatrix}. \end{aligned} \tag{33}$$

Polynomial entries in (33) have initial coefficients given by

$$c_0 = 1 \quad \text{and} \quad d_0 = k_1, \tag{34}$$

with remaining coefficients being recursively defined by

$$\begin{aligned}\mathbf{c}_N &= \begin{bmatrix}\mathbf{c}_{N-1} & 0 & 0\end{bmatrix}^\intercal + k_N \begin{bmatrix}0 & \tilde{\mathbf{d}}_{N-1} & 0\end{bmatrix}^\intercal \\ \mathbf{d}_N &= \begin{bmatrix}\mathbf{d}_{N-1} & 0 & 0\end{bmatrix}^\intercal + k_N \begin{bmatrix}0 & \tilde{\mathbf{c}}_{N-1} & 0\end{bmatrix}^\intercal,\end{aligned} \tag{35}$$

where $\tilde{\cdot}$ denotes the retrograde vector, one where the order of elements is reversed (e.g. by multiplying with the exchange, or backward identity, matrix of appropriate size), and where the length-$2N$ coefficient vectors have the form

$$\begin{aligned}\mathbf{c}_N &= \begin{bmatrix}c_0 & 0 & c_2 & 0 & \dots & c_{2(N-1)} & 0\end{bmatrix}^\intercal \\ \mathbf{d}_N &= \begin{bmatrix}d_0 & 0 & d_2 & 0 & \dots & d_{2(N-1)} & 0\end{bmatrix}^\intercal,\end{aligned} \tag{36}$$

with odd-ordered coefficients (even-numbered vector elements) being zero since polynomial entries in (33) have only even-ordered terms (corresponding to the one-sample propagation delay per section and between junctions). Consistent with (31), for a number of junctions $N > 1$, final coefficients are given by

$$c_{2(N-1)} = k_1 k_N \quad \text{and} \quad d_{2(N-1)} = k_N. \tag{37}$$

Coefficient vectors for $N = 1$ are obtained by (34):

$$\mathbf{c}_1 = \begin{bmatrix}c_0 \\ 0\end{bmatrix} = \begin{bmatrix}1 \\ 0\end{bmatrix} \quad \text{and} \quad \mathbf{d}_1 = \begin{bmatrix}d_0 \\ 0\end{bmatrix} = \begin{bmatrix}k_1 \\ 0\end{bmatrix}, \tag{38}$$

for $N = 2$, by (35) or directly from (37):

$$\mathbf{c}_2 = \begin{bmatrix}c_0 \\ 0 \\ c_2 \\ 0\end{bmatrix} = \begin{bmatrix}1 \\ 0 \\ k_1 k_2 \\ 0\end{bmatrix}, \quad \mathbf{d}_2 = \begin{bmatrix}d_0 \\ 0 \\ d_2 \\ 0\end{bmatrix} = \begin{bmatrix}k_1 \\ 0 \\ k_2 \\ 0\end{bmatrix}, \tag{39}$$

and for $N = 3$, by (35) (expanded for illustration):

$$\begin{aligned}\mathbf{c}_3 &= \begin{bmatrix}\mathbf{c}_2 & 0 & 0\end{bmatrix}^\intercal + k_3 \begin{bmatrix}0 & \tilde{\mathbf{d}}_2 & 0\end{bmatrix}^\intercal \\ &= \begin{bmatrix}1 \\ 0 \\ k_1 k_2 \\ 0 \\ 0 \\ 0\end{bmatrix} + k_3 \begin{bmatrix}0 \\ 0 \\ k_2 \\ 0 \\ k_1 \\ 0\end{bmatrix} = \begin{bmatrix}1 \\ 0 \\ k_1 k_2 + k_2 k_3 \\ 0 \\ k_1 k_3 \\ 0\end{bmatrix} = \begin{bmatrix}c_0 \\ 0 \\ c_2 \\ 0 \\ c_4 \\ 0\end{bmatrix} \\ \mathbf{d}_3 &= \begin{bmatrix}\mathbf{d}_2 & 0 & 0\end{bmatrix}^\intercal + k_3 \begin{bmatrix}0 & \tilde{\mathbf{c}}_2 & 0\end{bmatrix}^\intercal \\ &= \begin{bmatrix}k_1 \\ 0 \\ k_2 \\ 0 \\ 0 \\ 0\end{bmatrix} + k_3 \begin{bmatrix}0 \\ 0 \\ k_1 k_2 \\ 0 \\ 1 \\ 0\end{bmatrix} = \begin{bmatrix}k_1 \\ 0 \\ k_2 + k_1 k_2 k_3 \\ 0 \\ k_3 \\ 0\end{bmatrix} = \begin{bmatrix}d_0 \\ 0 \\ d_2 \\ 0 \\ d_4 \\ 0\end{bmatrix}\end{aligned} \tag{40}$$

and so on for models having a greater number of junctions.

With the scattering matrix $\mathbf{P}_N$ defined in (32) for cylindrical sections, substitution may be made into (24) to yield transfer functions in their more useful form, as a ratio of polynomial functions in z. The final expression for numerator and denominator polynomials are, however, dependent on the boundary conditions $R_0(z)$ and $R_L(z)$, whether they are scalar or frequency dependent and, in the latter case, the filter order.

4.1 $H_L(z)$ for Scalar Boundaries

In the simplified case (yet important because of its close relationship to LPC coefficients) of scalar boundaries, any losses may be lumped in R_0 and the reflection at the lips simply made lossless but inverting $R_L = -1$ (and thus no longer a function of z). Since, by (26), this would yield a transmission at the lips given by $T_L = 1 + R_L = 0$ and a complete attenuation of the signal, the transmission is omitted for this case. After a substitution of (32), the transfer function (24) therefore becomes

$$\begin{aligned}H_L(z) &= \frac{z^{-1}}{P_{1,1} + P_{1,2}R_L - R_0(P_{2,1} + P_{2,2}R_L)z^{-2}} \\ &= \frac{z^{-(N+1)} \prod_{m=1}^{N}(1 + k_m)}{K_{1,1} + K_{1,2}R_L - R_0(K_{2,1} + K_{2,2}R_L)z^{-2}} \\ &= \frac{B(z)}{A(z)},\end{aligned} \tag{41}$$

with the numerator being a pure delay with a scalar value,

$$B(z) = z^{-(N+1)} \prod_{m=1}^{N}(1 + k_m), \tag{42}$$

showing $H_L(z)$ has no zeros, and a denominator given by

$$\begin{aligned}A(z) &= K_{1,1} + K_{1,2}R_L - R_0\left(K_{2,1} + K_{2,2}R_L\right) z^{-2} \\ &= a_0 z^{-0} + a_1 z^{-1} + \cdots + a_{2(N+1)} z^{-2(N+1)},\end{aligned} \tag{43}$$

with polynomial coefficients given by the (column) vector

$$\mathbf{A}_N = \mathbf{C}_N \mathbf{R}, \tag{44}$$

where $\mathbf{C}_N$ is $(2N+3)\times 4$ matrix with columns constructed from coefficient vectors $\mathbf{c}_N$ and $\mathbf{d}_N$ given in (35),

$$\begin{aligned}\mathbf{C}_N &= \begin{bmatrix}\mathbf{c}_N & 0 & 0 & 0 \\ 0 & \tilde{\mathbf{d}}_N & 0 & 0 \\ 0 & 0 & \mathbf{d}_N & 0 \\ 0 & 0 & 0 & \tilde{\mathbf{c}}_N\end{bmatrix} \\ &= \begin{bmatrix}c_0 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & \mathbf{0} & \mathbf{0} \\ c_2 & d_{2(N-1)} & d_0 & \mathbf{0} \\ 0 & 0 & 0 & 0 \\ c_4 & d_{2(N-2)} & d_2 & c_{2(N-1)} \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ c_{2(N-1)} & d_2 & d_{2(N-2)} & c_4 \\ 0 & 0 & 0 & 0 \\ \mathbf{0} & d_0 & d_{2(N-1)} & c_2 \\ \mathbf{0} & \mathbf{0} & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & c_0\end{bmatrix}\end{aligned} \tag{45}$$

with each column extending the length-$2N$ vector by 3 zeros (in bold for better visibility) to accomodate a downward shift by one element from one column to the next, and where $\mathbf{R}$ is a 4×1 column vector

$$\mathbf{R} = \begin{bmatrix}1 & R_L & -R_0 & -R_0 R_L\end{bmatrix}^\intercal = \begin{bmatrix}1 & -1 & -R_0 & R_0\end{bmatrix}^\intercal \tag{46}$$

holding scalar boundaries R_0 and $R_L = -1$. It may be observed from (44)-(46) that the interleaved zeros characterizing vectors $\mathbf{c}_N$ and $\mathbf{d}_N$ carries over to the coefficient vector $\mathbf{A}_N$. Further, since by (34) $c_0 = 1$, the first element of $\mathbf{A}_N$ is $a_0 = 1$ and the last element is $a_{2(N+1)} = R_0$ so that the structure of the length $2N + 3$ coefficient vector is

$$\mathbf{A}_N = \begin{bmatrix} 1 & 0 & a_2 & \dots & 0 & a_{2N} & 0 & R_0 \end{bmatrix}^\mathsf{T}. \quad (47)$$

Equation (43) shows that for M cylindrical sections and $N = M - 1$ junctions using scalar boundaries R_0 and $R_L = -1$, the transfer function $H_L(z)$ is of order $2(N+1)$ and, save a scalar with pure delay in the numerator (42), an all-pole filter consistent with the assumption made in LPC of speech [4]. In fact, for cylindrical sections with simplified scalar boundaries, the coefficient vector $\mathbf{A}_N$ corresponds (save rounding error) to the autoregressive predictor coefficients estimated directly from the impulse response of $H_L(z)$ by LPC when the order is $2(N + 1)$ (and the unknown glottal flow and *true* boundary losses do not contribute to the observed signal and complicate the prediction—see Appendix 1).

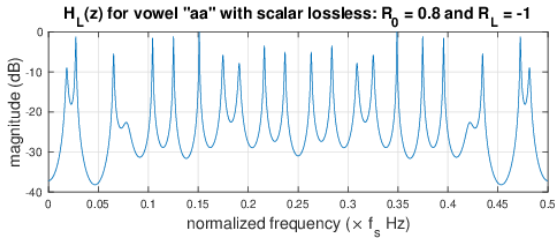


Figure 6. A scalar boundary loss, $R_0 = 0.8$ and $R_L = -1$ produces a symmetry in the half-bandwidth of the frequency response magnitude.

Factoring $\mathbf{A}_N$ for the scalar loss case show poles that are symmetric about the unit circle and a corresponding symmetry in the quarter-bandwidth (sampling rate divided by four) of the frequency response magnitude, as seen in Figure 6 for area functions of the vowel sound "aa" [11].

4.2 $H_L(z)$ for Frequency-Dependent Boundaries

In more practical applications of voice modeling, though the reflection at the glottis can often be approximated by a scalar, the reflection at the mouth is better modeled by accounting for frequency-dependent loss. Borrowing from work in which waveguide elements are estimated from measurement [12] and in which the open-end reflection of a cylindrical tube was shown to be very close to theoretical expectation [13], it was found here that a cascade of two first-order shelf filters, producing a second-order-section (SOS) and having the form

$$R_L(z) = \frac{B_L(z)}{A_L(z)} = -\frac{(b_L)_0 + (b_L)_1 z^{-1} + (b_L)_2 z^{-2}}{1 + (a_L)_1 z^{-1} + (a_L)_2 z^{-2}}, \quad (48)$$

with coefficient vectors

$$\begin{aligned} \mathbf{B}_L &= \begin{bmatrix} (b_L)_0 & (b_L)_1 & (b_L)_2 \end{bmatrix} \\ \mathbf{A}_L &= \begin{bmatrix} (a_L)_0 & (a_L)_1 & (a_L)_2 \end{bmatrix}, \end{aligned} \quad (49)$$

and with a transition frequency of $\omega_t = c/r_M$, as shown in Figure 7, produced a very good fit. With the assumption that the transmission is the amplitude complement of the lip reflection (26), it may be given here by

$$T_L(z) = 1 + R_L(z) = \frac{A_L(z) + B_L(z)}{A_L(z)}. \quad (50)$$

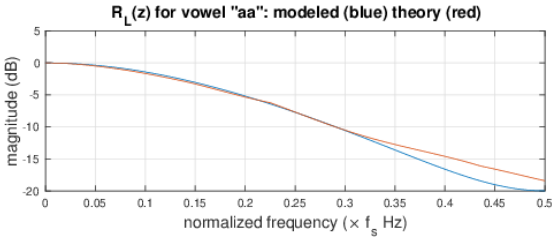


Figure 7. A cascade of two first-order shelf filters, each with band-edge gain of -10 dB, produces a second-order filter having the form given in (48) and with transition $f_t = c/(2\pi r_M)$ Hz (blue). This modeled response produces a good fit to the theoretical response of an open-end cylindrical tube having radius r_M, the radius of the final cylindrical section when the vocal tract model is configured to the vowel sound "aa" (red).

Substituting (48) and (50) into (41) yields the vocal tract transfer function tapped at the lips with frequency-dependent boundaries (denoted by $\hat{\cdot}$):

$$\begin{aligned} \hat{H}_L(z) &= \frac{T_L(z) z^{-(N+1)} \prod_{m=1}^{N} (1 + k_m)}{K_{1,1} + K_{1,2} \dfrac{B_L(z)}{A_L(z)} - R_0 \left(K_{2,1} + K_{2,2} \dfrac{B_L(z)}{A_L(z)} \right) z^{-2}} \\ &= \frac{\hat{B}(z)}{\hat{A}(z)}, \end{aligned} \quad (51)$$

where the numerator as a polynomial in z is given by

$$\begin{aligned} \hat{B}(z) &= (A_L(z) + B_L(z)) z^{-(N+1)} \prod_{m=1}^{N} (1 + k_m) \\ &= \left(b_0 + b_1 z^{-1} + b_2 z^{-2} \right) z^{-(N+1)} \prod_{m=1}^{N} (1 + k_m), \end{aligned}$$

having coefficients obtained by summing vectors in (49),

$$\begin{bmatrix} b_0 & b_1 & b_2 \end{bmatrix} = \mathbf{A}_L + \mathbf{B}_L, \quad (52)$$

and showing an introduction of zeros into the all-pole transfer function $H_L(z)$ given in (41) for scalar boundaries. The denominator of (51) as a polynomial in z is given by

$$\begin{aligned} \hat{A}(z) &= K_{1,1} A_L(z) + K_{1,2} B_L(z) - \\ &\qquad R_0 \left(K_{2,1} A_L(z) + K_{2,2} B_L(z) \right) z^{-2} \\ &= \hat{a}_0 z^{-0} + \hat{a}_1 z^{-1} + \cdots + \hat{a}_{2(N+2)} z^{-2(N+2)}, \end{aligned} \quad (53)$$

showing polynomial multiplication terms that require (acyclic) convolution of coefficients to produce coefficient vector $\hat{\mathbf{A}}_N$ which, in matrix form, is given by

$$\begin{aligned}\hat{\mathbf{A}}_N &= \begin{bmatrix}\hat{a}_0 & \hat{a}_1 & \hat{a}_2 & \dots & \hat{a}_{2(N+2)}\end{bmatrix} \\ &= \begin{bmatrix}\mathbf{C}_N \\ 0\,0\,0\,0 \\ 0\,0\,0\,0\end{bmatrix}\hat{\mathbf{R}}_0 + \begin{bmatrix}0\,0\,0\,0 \\ \mathbf{C}_N \\ 0\,0\,0\,0\end{bmatrix}\hat{\mathbf{R}}_1 + \begin{bmatrix}0\,0\,0\,0 \\ 0\,0\,0\,0 \\ \mathbf{C}_N\end{bmatrix}\hat{\mathbf{R}}_2,\end{aligned} \tag{54}$$

where $\mathbf{C}_N$ from (45) is extended by two rows of zeros to accommodate the convolution length $(2N+3)+2$ (the sum of $\mathbf{C}_N$ column length and $R_L(z)$ coefficient vector length minus one) and the downward shift of one row for each subsequent term in the sum. Equation (54) also requires a modification of the scalar boundary vector in (46) so it holds coefficients of $R_L(z)$ for corresponding n^{th}-order terms as indicated by the subscript n:

$$\hat{\mathbf{R}}_n = \begin{bmatrix}(a_L)_n & (b_L)_n & -(a_L)_n R_0 & -(b_L)_n R_0\end{bmatrix}^\intercal. \tag{55}$$

Finally, in addition to being two elements longer than $\mathbf{A}_N$, it is, perhaps, worthwhile to note that the coefficient vector $\hat{\mathbf{A}}_N$ no longer has the structure of interleaved zeros for odd-ordered terms, and is a vector more typical of an LPC estimation from an actual recorded speech signal.

5. DISCUSSION AND CONCLUSIONS

As mentioned in Section 4 and Appendix 1, the coefficient vector $\mathbf{A}_N$ corresponds to the linear prediction coefficients estimated from the impulse response of $H_L(z)$ if the LPC order is $2(N+1)$. If, on the other hand, the LPC estimation is on the impulse response of $\hat{H}_L(z)$ and, correspondingly, the order is increased to $2(N+2)$, the order of $\hat{\mathbf{A}}_N$, the estimated coefficients will not accurately correspond to $\hat{\mathbf{A}}_N$ since the assumption of an all-pole filter no longer holds. Though an increased order creates a better fit, this also introduces computational cost and, more significantly, impedes the inverse problem by placing the burden of representing losses on an increased number of reflection coefficients k_m, thus reducing their correlation to vocal tract length and cross-sectional area (parameters frequently estimated from LPC coefficients).

Another way of comparing the LPC estimation to the piecewise cylindrical waveguide model is by testing its (in)accuracy in the inverse problem of source-filter separation and glottal flow estimation. Though a rigorous treatment is beyond the scope of this work, a preliminary attempt at separating a model [8] generated volume flow (source) from the vocal tract model $H_L(z)$ presented herein (filter), can provide some insight into the role of the boundaries. Consistent with expectation, as shown in Figure 9, the inverse filter constructed with LPC estimated coefficients produces a signal (middle) that is closer to the signal produced by the inverse of $\hat{H}_L$ *without* the transmission filter $T_L(z)$ (bottom), a signal known as the flow *derivative*, frequently estimated and fit to the well-known parametric LF source model [14, 15]). When this flow derivative is passed through an inverse lip radiation

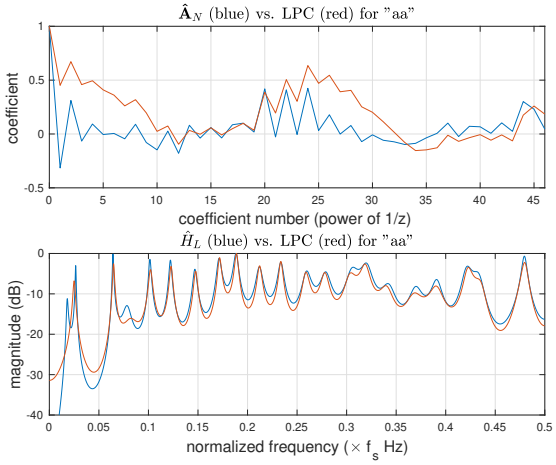


Figure 8. Coefficient vector $\hat{\mathbf{A}}_z$ and same-order LPC estimation from the impulse response of $\hat{H}_L(z)$ (top) and corresponding frequency response magnitudes (bottom).

(derivative) filter $L(z) = 1 - dz^{-1}$, where d is close to one ([6, 16]), the resulting signal (Figure 10, middle) is effectively integrated and shows a closer fit to the original volume flow (Figure 10, top), strongly suggesting that an accurate (acoustically-informed) lip reflection with amplitude-complementary transmission, may improve the problem of source estimation.

6. APPENDIX 1

For the transfer function

$$H_L(z) = \frac{Y_L(z)}{X(z)} = \frac{z^{-(N+1)}\prod_{m=1}^{N}(1+k_m)}{1+\sum_{i=1}^{2(N+1)} a_i z^{-i}}, \tag{56}$$

the difference equation is given by the inverse z-transform to yield output $y(n)$ at time sample n:

$$y(n) = \prod_{m=1}^{N}(1+k_m)x(n-(N+1)) - \sum_{i=1}^{2(N+1)} a_i y(n-i). \tag{57}$$

The impulse response $h(n)$ is the output in response to an input that is the unit step function $x(n) = u(n)$ which, by definition, has a non-zero value only when $n = N+1$, yielding

$$h(n) = \begin{cases} 0, & \text{for } n < N+1 \\ \prod_{m=1}^{N}(1+k_m), & \text{for } n = N+1 \\ -\sum_{i=1}^{2(N+1)} a_i h(n-i), & \text{for } n > N+1. \end{cases} \tag{58}$$

In linear prediction, future values of a discrete-time signal are estimated as a linear function of previous samples, a model that may be represented by an expression that is very similar to the final case of (58) where the impulse response $h(n)$ is defined for $n > N+1$ (an actual model

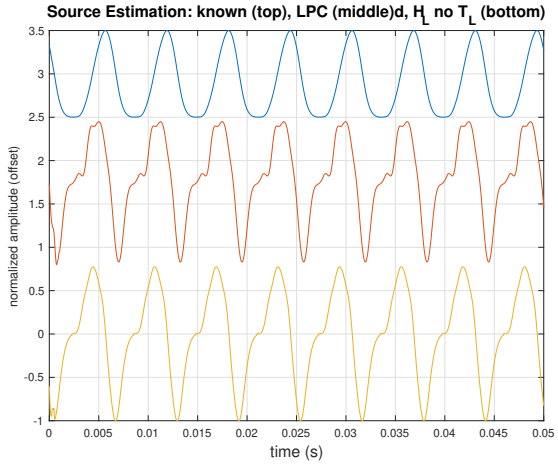


Figure 9. Known model glottal flow (top); signal from inverse filter with LPC estimated coefficients (middle); signal from $1/\hat{H}_L(z)$ *without* transmission $T_L(z)$ (bottom).

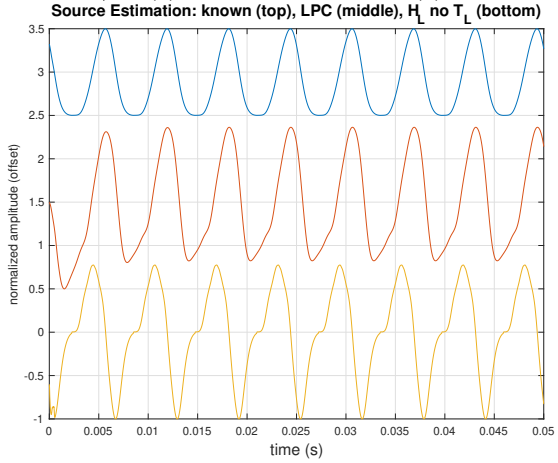


Figure 10. Known model glottal flow (top); signal from inverse filter with LPC estimated coefficients *with* the inverse of a lip radiation filter $1/L(z)$ (middle); signal from $1/\hat{H}_L(z)$ *without* transmission $T_L(z)$ (bottom).

would also account for prediction error). In practice, for a sufficiently long $h(n)$, one with enough samples that the infinite impulse response is allowed to decay very close to zero, the estimation of coefficients corresponding to $\mathbf{A}_N$ may be made with negligible error.

7. REFERENCES

[1] D. P. Berners, "Acoustics and signal processing techniques for physical modeling of brass instruments," Ph.D. dissertation, Stanford University, Stanford, California, July 1999.

[2] T. Smyth and F. S. Scott, "Trombone synthesis by model and measurement," *EURASIP Journal on Advances in Signal Processing*, vol. 2011, no. Article ID 151436, p. 13 pages, 2011, doi:10.1155/2011/151436.

[3] V. Välimäki and M. Karjalainen, "Improving the Kelly-Lochbaum vocal tract model using conical tube sections and fractional delay filtering techniques," in *Proceedings of the 3^{rd} International Conference on Spoken Language Processing (ICSLP)*, Yokohama, Japan, January 1994.

[4] J. D. Markel and A. H. Gray, *Linear Prediction of Speech*. Springer-Verlag, 1976.

[5] J. Jr and C. Lochbaum, "Speech synthesis," *Proceedings of the fourth International Congress on Acoustics*, vol. G42, pp. 1–4, 1962.

[6] O. Perrotin and I. McLoughlin, "A spectral glottal flow model for source-filter separation of speech," in *Proceedings of 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK, May 2019.

[7] J. O. Smith, *Digital Waveguide Modeling of Musical Instruments*. ccrma.stanford.edu/~jos/waveguide/, 2003, last viewed 12/4/08.

[8] T. Smyth and A. Fathi, "Voice synthesis using the generalized pressure controlled-valve," in *Proceedings of ICMC 2008*, Belfast, Ireland, August 2008, pp. 57–60.

[9] J. O. Smith, "Physical audio signal processing for virtual musical instruments and audio effects," http://ccrma.stanford.edu/~jos/pasp/, December 2008, last viewed 8/24/2009.

[10] H. W. Strube, "Are conical segments useful for vocal tract simulation?" *Journal of the Acoustical Society of America*, vol. 114, no. 6, pp. 3028–3031, December 2003.

[11] B. H. Story and I. R. Titze, "Vocal tract area functions from magnetic resonance imaging," *Journal of the Acoustical Society of America*, vol. 100, no. 1, pp. 537–554, July 1996.

[12] T. Smyth and J. Abel, "Estimating waveguide model elements from acoustic tube measurements," *Acta Acustica united with Acustica*, vol. 95, no. 6, pp. 1093–1103, 2009.

[13] H. Levine and J. Schwinger, "On the radiation of sound from an unflanged circular pipe," *Phys. Rev*, vol. 73, no. 4, pp. 383–406, 1948.

[14] G. Fant, J. Liljencrants, and Q. Lin, "A four-parameter model of glottal flow," *Royal Institute of Technologies - Dept. for Speech, Music and Hearing, Quarterly Progress and Status Report 4*, 1985.

[15] H.-L. Lu and J. O. Smith, "Joint estimation of vocal tract filter and glottal source waveform via convex optimization," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'99)*, New Paltz, NY, October 1999, pp. 79–92.

[16] I. V. McLoughlin, *Speech and Audio Processing: a MATLAB-based approach*. Cambridge University Press, 2016.

COMPOSING WITH FEEDBACK

Silvia LANZALONE (info@silvialanzalone.it)[1, 2]

[1]**CRM - Centro Ricerche Musicali**, Rome, Italy
[2] **Latina State Conservatoire of Music 'Ottorino Respighi'**, Latina, Italy

ABSTRACT

The paper illustrates my compositional research aimed at the creation of musical works distinguished by the integration of computer, electronic and electroacoustic systems with instruments, or other acoustic-mechanical vibrating bodies, for the augmentation of musical sound and gesture through the feedback phenomenon.

The use of acoustic phenomena for compositional purposes, such as resonance, reflection, feedback, is part of a research area exploring the possibility of a tangible acoustic presence in the listening space.

The first part of the paper deals with historical and aesthetic considerations concerning on the creative use of feedback in music. Some previous research on augmented instruments embedding feedback systems is then presented. These studies have been carried out since the 1990s by the CRM - Centro Ricerche Musicali in Rome, where I designed and built my works and related systems.

The paper then focuses on my recent feedback augmented instrument ResoFlute, designed for the work *Èleghos*, for augmented flute, resonant pipe, and electronics (2014), and *Feedback for Two*, for 2 voices, 2 megaphones, and 2 feedback systems (2016-2018).

1. THE AESTHETICS OF FEEDBACK

The word "feedback" indicates a wide category of phenomena characterized by the effect of the action of a dynamic system on the system itself, up to the point of modifying its results. Feedback is used not only in electronics and computer science, but also in many other fields, such as biology, neurology, psychology, linguistics, acoustics, music. Feedback is essential in all traditional musical applications, from instrumental practice, including performance, interpretation, and improvisation, up to composition.

In the compositional field, the feedback was intended and produced in different ways, in the analogue and digital domain, starting from the temporal control of musical sound, up to signal processing. In *Solo* (1966) by Karlheinz Stockhausen, and in *I Am Sitting in a Room* (1969) by Alvin Lucier, for example, main compositional process consists of two different feedback systems based on tape delay, whereas in *Post-prae-ludium per Donau* (1987) by Luigi Nono, feedback delays are used in different types of digital algorithms, to create both polyphonic structures and sound processing. In art music, from the 1960s onwards, several composers have been able to use the principle of controlled audio feedback in their works, using generic or specific electroacoustic systems, often in conjunction with computer systems in more recent works. Examples of historical and recent compositions include, among others: *Electronic Music for Piano* (1964) by John Cage; *Pendulum Music* (1968) by Steve Reich; *Microphone* (1973) by David Tudor, *Pea Soup* (1974) by Nicolas Collins, *Bird and Person Dyning* (1975) and *Empty Vessels* (1997) by Alvin Lucier, *Wings* (2007-08) by Cathy van Eck, the cycle *Modi di interferenza* (2006-2010) by Agostino Di Scipio.

The development of computer systems has made possible the implementation of mathematical models for the composition of sound and music. Xenakis' insights into the application of the Gabor's model on the elementary acoustic signal, for example, bring the morphological correspondences of sounds with natural processes into music. Such correspondences are possible through the creative application of physical and mathematical theories, such as the kinetic theory of gases, or statistical theories, whose calculations require the use of computers [1]. The computer allows to create models or processes of dynamic composition of music, both in terms of sound and structural relationships between sounds. The computer becomes a real thinking 'tool', capable of establishing a feedback fast enough to ensure dynamic interaction between man and machine [2]. Feedback techniques are now often used for the implementation of self-generating systems, up to the most recent creation of music with adaptive features, as in the work of Michelangelo Lupone [3, 4]. Adaptivity is in fact possible when the system can store previous states and to react to external stimuli based on the content stored in the memory.

The interest of researchers and composers in the production and processing of sound, first analog and then digital, leads to the use of feedback for the implementation of filters and delays, at the basis of many complex digital algorithms. The use of feedback in audio digital processing also includes synthesis by physical models, developed between the 80s and 90s. See, for example: J. O. Smith and P. Cook's studies at Stanford University's CCRMA; Modalys, physical modeling synthesis program developed at IRCAM; Physical modelling synthesis of bowed strings, by L. Seno e M. Palumbi, developed at CRM-Centro Ricerche Musicali [5, 6].

Research was aimed not only at imitating instrumental sound, but above all at its reinvention, to create new virtual models. Therefore, also in this case, "speed" is fundamental, because allows the computer to perform calculations fast enough to simulate "real time" and, with suitable control tools, allows the user to "play the machine" as a musical instrument. The creation of virtual instruments is therefore also connected to the production of specific controllers, or interfaces for controlling digital algorithms. The interfaces are ergonomically designed to achieve speed and precision for calibrated data processing, as well as comfort and naturalness of gesture for expressive music performance. Research on the improvement of tactile feedback, by using force-feedback on haptic technologies, aims to make the gesture musically effective, bringing it closer to the gesture used on traditional instruments. A category of interfaces, called "hyperinstruments", picks up the articulatory characteristics of the instrumental gesture directly on the instrument itself, during performance. Some hyperinstruments are, for example: MIT's Media Lab Hyperstring Project (1990-93); STEIM Overtone Violin (2004); IRCAM Augmented Violin (2003-2006); InfoMus Lab Iperviolino (2008); STEIM Meta-trumpet (1994) Softwind MIDI Synthophone (1986); McGill University's CIRMMT Hyper-flute (1999); ICST SABRe-Sens or Augmented Bass Clarinet Research (2012); KMH of Stockholm's Hyper-Hurdy-Gurdy (2015); CCRMA's Hybrid Lutheries Project (2018). To enable the performer himself to control the electronic part, sensors are applied inside, above, and around the instrument, or on the body of the performer. The sensors detect the gestures normally used to play, or the ancillary gestures that occur while playing, or even additional utilitarian gestures, specifically identified by the composer. The ergonomic and technological choices of the type of control or sensor, the decoding of the data and the mapping of the parameters, contribute to the expressive characterization of the music. The gesture can be considered in an ergonomic / instrumental context, if linked to physiology, or a compositional / musical context, if linked to sound. The difficulty of playing an instrument is directly proportional to the possibility of the instrument itself to respond appropriately to the complexity of our physical system. However, the extension of the "gesture" does not completely solve the augmentation of the instrument, which uses computer and electro-acoustic technologies inside its vibrating body, to increase generation, processing, and diffusion of the sound. The substantial dissociation between the production of sound and music and its propagation is a condition common to a large part of electroacoustic music production. In music for instruments and live electronics, the sound diffused by the loudspeakers remains unrelated to the instrumental sound, even with respect to perceptual identification in the space of its diffusion. Electronic sound, whether instrumental, concrete, synthetic, or processed, is perceived as 'disembodied', because it is presented in a 'virtual' context, separated from the perceptual reference of the body that causes, generates, and spreads acoustic vibrations. The dissociation between the generation of music and its propagation in the space of performance is common to many contemporary music, from the first works of acousmatic music to the so-called "liquid music". The need to restore the close connection between the vibrating body and the related acoustic phenomena is intertwined with the aesthetic need to manipulate the sound material. The research area dealing with augmented instruments has a strong point in the integration of electro-acoustic and digital technologies to the mechanical structure of the acoustic instrument [7].

Feedback has been widely studied and implemented by both researchers and electronic component industries, for example in operational amplifiers and in power amplifiers (negative feedback)[1], or in oscillating circuits (positive feedback)[2], which were the basis to produce many electronic instruments and sound processing systems. Some of these are, for example: Synket, o Syn-ket, o Synthesiser-Ketoff, by Paolo Ketoff, (1963); Moog, o Moog Synthesiser, by Robert Moog (1964); Chua oscillator, o Chua circuit, by Leon O. Chua (1983).

The instrumental sound comes from the close interaction between exciter, resonator, and radiator. In "solid body" electric guitars and violins, the contribution of the sound box is replaced by an analogue or digital sound processing system and by an electroacoustic amplification system. The first electric guitar, called "frying pan", by Georges Beauchamp (1934), produced by Adolf Rickenbacker, was immediately followed by the production of "solid body" guitars, such as the Telecaster and Stratocaster models by Fender, or Les Paul by Gibson, introduced in the 1950s and still in production; the first electric violin prototypes were mainly acoustic instruments with contact microphones inside the sound box, such as the Giant-Tone Radio Violin by R.F. Starzl (1927), or electromagnetic pickups placed under the bridge, often without a sound box, as VioLectric, by Fredray H. Kislingbury (1939). Traditional sound box amplification is the cause of "annoying" feedback and its elimination allows the sound of the strings to be picked up directly, making the system much more stable. The acoustic phenomenon known as "Larsen effect"[3] is in fact considered an unwanted "accident", to be avoided, or minimized, through the production and use of specific hw/sw technologies, especially in the professional audio sector [7].

In the 1960s, Jimi Hendrix also produced feedback for expressive purposes by bringing the pickups of the electric guitar closer to the amplifier cone, consolidating a musical practice that was spreading into rock and jazz, and more generally into the context of pop music. The need to prolong the sound, has also stimulated the development of several inventions, some of which use feedback, as well as other properties of electromagnets. Some experiments were done on the piano, resulting in early prototypes, such as: Electrophonischen Apparat (1886) by Richard Eisenmann; Electrochord and Neo-Bechstein, by Oskar

[1] Feedback is defined as negative when output attenuation and system stabilization occurs.

[2] Feedback is defined as positive when output increases exponentially, up to the instability of the system.

[3] Positive feedback discovered by Søren Absalon Larsen (1871–1957).

Vierling in the 30s, at Heinrich Hertz Institute in Berlin. Some research was also done on the electric guitar, resulting in some prototypes still used today, such as: E-Bow (1969) by Greg Heet; Guitar resonator, of Vibesware Company, Fernandes system, patented in the 90s; Moog Guitar (2008) by Paul Vo. An important step in the research on augmented instruments was given by the possibility of implementing audio feedback in combination with digital signal processing systems that allow, with appropriate signal conditioning, to modify the structure of the feedback and to manipulate the sound. See, for example: Metasaxophone, by Matthew Burtner, at Stanford University's CCRMA (1997); Magnetic Resonator Piano, by Andrew McPherson, at Centre for Digital Music of Queen's Mary University of London (2010); Halldorophone, by Halldór Úlfarsson at EMuTe Lab - Experimental Music Technologies Lab, University of Sussex, UK (2018) [7].

The use of controlled audio feedback for the creation of augmented instruments, and for the creation of original musical works, is part of a research area that has only recently taken on relevance in musical thought, resulting in several interesting inventions and various musical pieces, in many electronic music research and production centres [8, 9].

Specific studies on feedback have been carried out also by CRM - Centro Ricerche Musicali in Rome, since the 1990s. The research carried out by CRM is constantly guided by aesthetic principles and pursues the objective of creating works and technologies aimed at developing the musical language and creating an innovative artistic production, resulting from the constant interaction between scientific research and musical composition [10].

2. PREVIOUS RESEARCH AT CRM

The restitution of a "corporeality" to sound and its ability to offer a "tangible" presence in the listening space is achieved through experiments on the vibrational properties of matter and other acoustic phenomena studied by the CRM of Rome since 1994.

Many interactive and adaptive sound installations have been designed with waveguides, resonators, reflective screens, multiphonic systems, or with original technologies such as Holophones and Planophones®, both by Michelangelo Lupone.

Holophones (CRM, 1998), sound projectors, are the first prototype of a multiphonic sound diffusion system provided with very accurate controls, which permit creative modulations of the wavefront. Holophone consist of a parabolic surface with a limited band loudspeaker in its focal point with controllable radiation angle. Based on plane waves emission, the Holophone is designed to ensure an accurate control of the soundwave movement and profile by appropriate regulation of the phase, amplitude, and frequency of the musical signal [11, 12].

Planephones® (CRM, 1996) are vibrating plates that allow exploiting the vibrational features of natural and synthetic materials, such as metals, wood, paper, glass, and their derivatives, with musical and plastic form. They can be considered both planar sound radiators that allow diffusion of plane waves, and instruments for sound elaboration and spatial diffusion when integrated into art sound installations [13].

These projects, together with previous studies on the physical model of the string [5, 6], have contributed to the development of research on augmented instruments produced at CRM, as well as to the deepening of my personal compositional research.

2.1 Sculptures

Experiments on resonant cavities and vibrating panels of different shapes and materials led me to create musical installations in collaboration with the visual artist Debora Mondovì, as part of a research started in 2005 on the sounds that can be produced with terracotta. Terracotta is extremely ductile and can be moulded into a variety of shapes using different techniques. The terracotta sculptural works have been designed considering the relationship between the material, the acoustic phenomenon, and the electroacoustic system, to achieve a complete integration between sculptural and musical form [14].

Risonanze dalla Terra (2007), *Terra delle Risonanze* (2010) and *Voci di Terra* (2011) were designed to resonate at certain frequencies and consist of sculptural elements used as resonators, where amplification, or attenuation phenomena occur, according to the shape and the size of the cavities.

Foglie di Terra e di Suono (2009) and *Voci d'amore* (2914) are instead made with terracotta panels of different sizes, shapes, and thicknesses, subjected to different cooking techniques. The different characteristics of the material and the different application of piezoceramic, or electromagnetic actuators, according to the Planofoni® technique, allows the creation of diversified sound elaboration and spatial diffusion, coherently with structures in terracotta.

2.2 Instruments

Specific applications of the resonance phenomenon on traditional acoustic instruments, gave rise to the production of three performative works, created from 2005 to 2011. In these works, the design of the electroacoustic system involved in the processing and diffusion of sound was integrated into the traditional acoustic instrument, or a stylized version of it.

In the work *Il suono incausato*[4] the clarinet is deprived of the mouthpiece and has a loudspeaker inserted in a soundproof cone frustum, inserted at the barrel height. This device carries out the function of conveying sound and air, produced by the movement of the loudspeaker membrane inside the instrument. The sound to the loudspeaker is produced by a real time algorithm made with the software Max/MSP and controlled by the performer – the clarinetist – through a MIDI pedal. The

[4] Silvia Lanzalone, *Il suono incausato*, improvise-action for suspended clarinet, clarinetist, and electronics. Editions Suvini Zerboni. Clarinetist Massimo Munari, Ivrea, Italy, 2005.

clarinet, thus modified, is suspended on a support, to give the clarinetist the possibility to "explore" the system, performing a gestural score. The score also indicates the gestures to be performed for the use of corks and trunks of cardboard or aluminum cones of different sizes, to be inserted respectively into the holes and into the bell, and of shims to be placed under the key levers [14, 15].

In the work *Voce*[5] the electronic part is performed through a structure made up of fifteen resonators, consisting of glass cavities of different shapes and sizes with fifteen small speakers, grouped on five stands. Resonators are selected to achieve the resonances of vowel sounds, and the signal processing algorithm emphasizes the resonances and creates further transformations consistent with the vocal sounds. The installation, called "glass choir", dialogues with the vocal performer, giving to the voice timbre an original and coherent sound characterization [14].

Clavecin électrique[6] is a performative work produced following research into the acoustic and technical possibilities of the historical harpsichord, with the aim of enhancing its sound diffusion and increasing its range of timbres. The work involves the integration of eight resonant pipes of different sizes, tuned to the main formants of the instrument. The pipes are placed below the soundboard, with their radiating end close to the main radiation area of the instrument. The sound of the harpsichord emitted by the pipes after digital processing, is thus transmitted to the soundboard, causing the sympathetic vibration of some strings [16].

The performative character common to the three works is developed in each of them according to different levels of freedom. The need to dramatize the gesture is implicit in the performance score for the new instrument. The performer can interact with the augmented instrument in an extemporaneous way, relating to the microstructural variations of the sound within the formal units; the timing, that is the choice of the duration of each formal unit; the management and morphology of theatrical actions.

3. FEEDBACK INSTRUMENTS AT CRM

Several prototypes of augmented instruments, realised with electroacoustic and computer systems in which the feedback phenomenon is involved, have been produced at the CRM laboratories: Feed-Drum®, SkinAct and WindBack, by Michelangelo Lupone; ResoFlute, by Silvia Lanzalone [7, 10].

The Feed-Drum®, designed in 1999 by Michelangelo Lupone for his work *Gran Cassa* (1999-2002), in order to explore the timbre during the bass drum attack phase and isolate its vibration modes by means of the membrane's electronic conditioning system. The signal produced by an excited membrane is returned through a speaker which, taking advantage of the feedback principle can potentially generate everlasting sounds. The damping of membrane motion leads to sound decay. The input energy level can be dynamically adjusted, in order to isolate high vibration modes [7, 10, 17, 18].

The SkinAct is an augmented membrane instrument designed by Michelangelo Lupone for the work *Spazio curvo* (2012), which was included in the work *Coup au Vent*, for three SkinActs (2015). The instrument can produce long sounds by exciting the membrane of a bass drum in a vertical position. The skin puts a vibrational detector and different actuators in feedback condition [7, 10, 18].

The WindBack, designed in 2011 by Michelangelo Lupone for his work *In sordina* (2011), is a saxophone modified by applying a loudspeaker in front of the bell and a miniature microphone placed near the mouth. The loudspeaker sends the sound of the instrument picked up by the microphone inside the instrument itself, causing acoustic feedback. The internal resonances allow the feedback to be tuned, and to modify its timbre in relation to the position of the keys [7, 10, 19].

3.1 ResoFlute

The personal research on the augmentation of the traditional instruments continues, between 2014 and 2018, with the production of two more performative works, in which the focus shifts to the musical and expressive properties of acoustic feedback.

ResoFlute, augmented instrument designed by myself for the work *Èleghos*[7], is a classical transverse Western concert flute, modified through the application of six miniature microphones inside the body of the instrument, a sensor, some control pedals, and an aluminum pipe for sound diffusion and resonance [Fig 1]. A miniature microphone is applied internally to the instrument's headjoint, instead of the traditional cork, and five other electret miniature microphones are placed on the body, making holes in the pipe according to the position of certain keys[8]. The six microphones detect resonances and sound pressure variations that can be found in different parts of the flute opening and closing the different keys. The body of the instrument also features a piezo-film sensor, used for detecting the position of the right-hand thumb, which is normally used by the flautist just to hold the instrument. The piezo-film signal is used via threshold circuit to produce on-off commands that the performer generates moving the right-hand thumb according to the score. The flute player also uses MIDI foot controls featuring Control Change and Program Change messages to activate certain sound processing according to the score. The sound of the traditional instrument is 'augmented' not only using microphones, but also through a digital signal processing algorithm and by the aluminum pipe. The latter

[5] Silvia Lanzalone, *Voce*, for small choir of glass, female voice, and electronics. Editions Ars Publica. Voice Angelina Yershova, Rome, Italy, 2006; Silvia Schiavoni, Rome, Italy, 2006.

[6] Silvia Lanzalone, *Clavecin électrique*, for harpsichord with resonant pipes and electronics. Editions Ars Publica. Harpsicord Giorgio Spolverini, Salerno, Italy, 2011; Rome, Italy, 2011.

[7] Silvia Lanzalone, *Èleghos*, for augmented flute, resonant pipe, and electronics (2014). First performance Festival ArteScienza 2014, Giardini dell'Accademia Filarmonica Romana, Rome, Italy, 06.07.2014. Flutist: Gianni Trovalusci. The previous version of the piece, entitled *Studio su Èleghos* was performed at the Auditorium of the University of Rome "Tor Vergata", Rome, Italy, 18.06.2014.

[8] The project on microphones was undertaken with the participation of Antonio Marra, audio technician and flute repair specialist.

not only diffuses the sound of the flute, but also filters it to enhance frequencies corresponding to its normal modes. The aluminum pipe mainly allows, thanks to its resonances and radiation directivity, to obtain controlled feedback and to tune the instrument without insufflation. The resonant aluminium pipe, 10 cm diameter and 180 cm length, is mounted inside a wooden case featuring the speaker and a second cabinet containing other electronic devices, including a power amplifier. The speaker is connected to the pipe through a conical joint, which enhances acoustic power effectiveness.

The algorithm, implemented with Max/MSP, consists of two banks of resonant filters 'tuned' on the pitches of the instrument. The microphone signal is analysed to extract amplitude and pitch envelopes, while the piezo film sensor signal is analysed to extract the amplitude peaks. Between the two filter banks are placed an FM synthesis process, a comb filter, and a modulation by means of a noise generator, all regulated by the analysis values. A compression algorithm is used to control acoustic feedback between the pipe and the microphones placed into the flute.

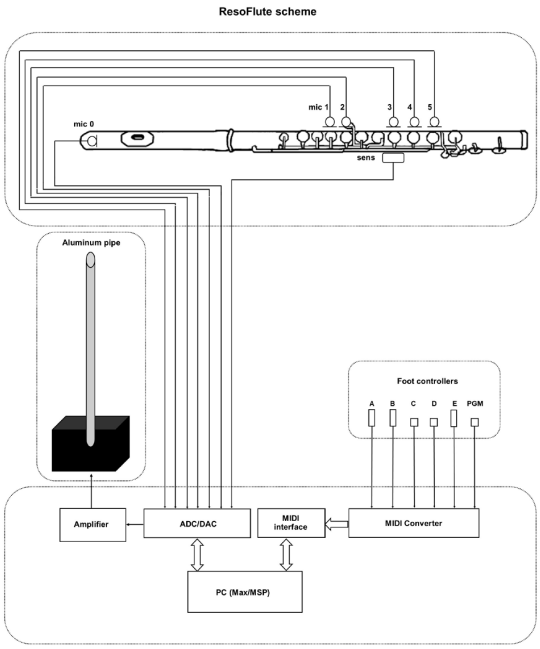


Figure 1. ResoFlute scheme.

Therefore, the entire system comprises an electroacoustic system for controlled feedback and digital algorithms for for generation and real-time processing of flute and feedback sounds. Feedback depends on resonant pipe features, responsible for sound diffusion, and on flutist position relative to the pipe. Feedback is produced and 'tuned' by specific gestures of the flutist, who modifies the sounds by moving the flute closer or further away from the pipe. The system has been designed to achieve maximum integration and balance between the natural sounds of the instrument and the electronic processing, both in terms of spectral and spatial diffusion [7, 10, 19].

The ResoFlute project is still under development for the construction of a new prototype, in collaboration with the architect and designer Emanuela Mentuccia[9]. The new prototype includes a system of wireless microphones and sensors inserted in a removable and easy-to-use structure, with a functional and non-invasive design, that can be applied to the traditional transverse flute without drilling holes or altering its structure.

Figure 2. Gianni Trovalusci plays ResoFlute. Roma, 2014.

4. FEEDBACK FOR TWO

Feedback for Two, composed between 2016 and 2018[10], experiments the possible relationships between two voices and two feedback systems, using two megaphones. The two megaphones are configured to process voice sounds and are placed at the center of two feedback processes, involving several transducers, internal and external to the same megaphones.

The typical exponential shape of the megaphone is a characteristic of horn loudspeakers, whose feature to adapt the acoustic impedance between the transducer and the surrounding air makes it possible to achieve a high-power

[9] The first phase of the project was included in the "Make Your Idea 2017" programme, dedicated to the development of new projects in the Lazio Region's FabLab network.

[10] Silvia Lanzalone, *Feedback for Two*, for 2 voices, 2 megaphones and 2 feedback systems (2016-2018). First version: EmuFest 2016, Rome Conservatoire of Music 'S. Cecilia', 27.10.2016. Final version: Festival ArteScienza 2018, Auditorium "Parco della Musica" of Rome, 14.09.2018. Voices and megaphones, Eleonora Claps and Virginia Guidi. Scene and lighting design Emanuela Mentuccia.

output. The use of the exponential horn for voice amplification, already practiced in Greece in the sixth century BC, is found among the inventions of Athanasius Kircher, a German Jesuit and scientist of the 17th century. Horn shape has always been used for acoustic amplification and was integrated in many inventions for sound reproduction in the late 19th and early 20th century. The horn was inserted, for example, into various models of phonographs (Thomas Edison, 1877), of graphophones (Chichester Bell and Charles S. Tainter, 1885), and gramophones (Emile Berliner, 1887), to help amplification of the reproduced sound. In 1899, J.M. Augustus Stroh patented a violin amplified by a mechanical system consisting of an elastic membrane and a horn, like those of the first mechanical phonographs and gramophones of the time, with the aim of being able to record the sound with sufficient volume. The instrument does not need a sound box but is composed of all the other parts of the violin, with the vibrating membrane placed near the bridge. This invention was later applied to other stringed instruments, including the viola, mandolin, and guitar, but was destined to disappear from the 1930s onwards, when recording by electric amplification became widespread [7]. In the Futurist Manifesto *L'arte dei Rumori* (The Art of Noises) of 1013, Luigi Russolo describes the "intonarumori", the instruments he created, all characterised by the presence of acoustic horns, whose purpose is to amplify the sound[11] [20]. Stroh's violin and Luigi Russolo's "intonarumori" are among the first examples of the implementation of the acoustic horn for musical purposes, outside the traditional musical instruments. In July 1951 Shitetsu Kamimori registered the patent for the "electric megaphone", a prototype of an electroacoustic megaphone with shape and technical characteristics like the current one[12].

The electric megaphone is functional for remote voice amplification and is normally considered as a "closed" system, not to be used in a traditional musical context. In the 1960s, however, the megaphone began to be used in original musical works, mostly experimental, and especially together with other instruments, although with a marginal role. Ligeti's works *Aventures*, for three singers and seven instrumentalists (1962), involves the use of 3 megaphones for the amplification of a single long sound of the voices that mixes with the sound of the ensemble. Kagel's work *Acustica*, for experimental sound generators and loudspeakers, composed between 1968 and 1970, involves 2 to 5 performers and uses many objects, recycled or self-made, some traditional instruments, such as a trumpet, a trombone, a violin, and various technologies for recording, reproduction, and sound amplification, such as turntables, microphones, speakers, including megaphones.

The use of the megaphone for musical composition in more recent times is still quite rare. There are very few works that include the megaphone in the ensemble, however always together with traditional instruments and with a coloristic function. In the work *Dead City Radio. Audiodrome*, for orchestra (2003), Fausto Romitelli uses the megaphone to amplify the voice of the percussionists. A recent attempt to bring the essential, almost sinusoidal timbre of the feedback that can be produced with the megaphone, into dialogue with instrumental sound, was made by Simon Steen-Andersen in 2008, with the work *On And Off And To And Fro*, for vibraphone, saxophone/Eb clarinet, double bass/cello and three players with megaphones. The work includes an electric megaphone model with a mobile microphone[13] for the pitch of acoustic feedback. The score provides detailed instructions on the placement of megaphones to amplify the instruments, or to produce the desired feedback.

The first work in which the megaphone plays a "solo" role seems to be *Solo per megafono,* by Giuseppe Chiari, whose score reads, in perfect "fluxus" style, <<suonare liberamente>>[14], without any other indication. The performances of *Solo per megafono* which took place in '68 in Palermo and Florence, seem to have been of considerable influence[15] [21]. In 2009 Alessio Rossato produced a work for megaphone quartet and conductor, entitled *Omaggio a Giuseppe Chiari*[16]. In the quartet, the fluxus approach is taken up in a structured way through specific indications in the score with respect to "sound acts" (irregular rhythms with the power button, whispering, breathing noisily, provoking feedback with the mouth next to the microphone), "gestural acts" (frontal motionless, turning left and right, megaphone up, turning around oneself) and other directions for playing with megaphones[17].

In my work *Feedback for Two,* I have chosen to use the megaphone as an "instrument" for musical performance, to take advantage of its electroacoustic and ergonomic characteristics for the controlled management of feedback in the air. The high acoustic efficiency of the megaphone allows to achieve, with minimally invasive components, a significant amplification at medium-high frequencies, easily to control for the management of the feedback phenomenon. The remarkable directivity of the sound radiation allows a very detailed selection of the feedback triggering space. Furthermore, the megaphone is already ergonomically optimized, not only to be held with one

[11] <<Gli intonarumori hanno esternamente la forma di una scatola più o meno grande a base generalmente rettangolare. Dal lato anteriore esce una tromba che serve a raccogliere e rafforzare il suono-rumore.>> (Luigi Russolo, 1913) [20]. Translation: <<The "intonarumori" have externally the shape of a more or less large box with a generally rectangular base. From the front side comes out a horn that serves to collect and strengthen the sound-noise>>.

[12] Official Gazette of the United States Patent Office, Volume 687, October 26, 1954. Shitetsu Kamimori's patent is dated July 13, 1951, and has serial number 236,498.

[13] The score indicates that megaphones must be "Velleman M25 SFM" (ed. Edition·S – music¬sound¬art Copenhagen, DK).

[14] Translated: <<To play freely>>. The score *Solo per megafono* (1968) is published in G. Chiari, *Musica Madre*, Giampaolo Prearo Editore, Milan 1973-2000, together with *Solo per clarinetto* (1964). Another "solo" by G. Chiari is *Solo per tromba*, Florence 1998, self-published. See: Mario Chiari, Fabio Migliorati, *Giuseppe Chiari, una bibliografia,* http://www.giuseppechiari.eu/.

[15] The score of *Solo per megafono* was previously included in the collection of G. Chiari entitled *La strada*, per objects, instruments, 1965 [21].

[16] A. Rossato has performed / interpreted *Solo per megaphone* by G. Chiari on several occasions, starting from 2009.

[17] Another piece by A. Rossato in which the megaphone is included in a mixed ensemble is *Ancient dark scarecrow*, for piano with carillon, 4 puppets, soprano with gong, speaking voice with megaphone and resonator, live electronics and 8 digitised tapes (2007).

hand and moved according to the acoustic space, but also to be manually activated and adjusted via a switch and a potentiometer. These features were very useful for controlling feedback in the air during the performative action required in the piece.

The megaphone model used in *Feedback for Two* has a power output of 10 Watts RMS, a receive range of about 500 meters and a weight light enough to play without excessive wrist fatigue[18]. The technical adjustments to the megaphones were the exclusion of the original microphone, which is band-limited and placed in a fixed cabinet behind the horn, to avoid feedback. The application, in place of the original microphone, of a professional miniature microphone with omnidirectional characteristics inserted directly into the horn, required a slight acoustic correction. Sound-absorbing material was in fact placed at the points of maximum acoustic efficiency, where the reflections on the horn walls emphasize the feedback, making it unstable. Each megaphone system has two feedback chains: the first between microphone and megaphone and the second between megaphone and loudspeaker.

The first of the two feedback in each of the two systems used by the two vocal performers, takes place between the omnidirectional miniature microphone[19] and the loudspeaker of the megaphone. The second feedback involves a second directional miniature headset microphone[20], placed in front of each performer's mouth. A third feedback occurs between one of the two microphones - the megaphone microphone, or the performer's microphone - and one or more loudspeakers, in a different way during the piece, according to the score. The speakers used have high directivity[21] and were placed in the middle of the stage, at eye level and behind the stage action, to allow the performers to approach and trigger the controlled feedback required by the score. The speakers contribute to the activation of the feedback and have the function of spreading the sound in the hall, so the work is self-sufficient in terms of its diffusion in the listening space[22]. Megaphones are also a very efficient acoustic source, if suitably pointed towards the audience by the two performers, as happens in some points of the work. Each described feedback is subjected to different signal processing within the feedback loop, with a latency of about 12 milliseconds. The live electronics algorithm, implemented with Max/MSP and controlled by the performer at computer, is composed of different processing stages, including compressors, delay with feedback, resonant filters, pitch detectors and pitch shift. Digital processing of the feedback signal is fundamental, not only for its dynamic control, but also to be able to obtain as many timbre variations as possible, such as harmonic tuned spectra, colored noise, short granular sounds, long glissando sounds, clusters with internal beats, and various other sounds. The performer at computer controls the feedback via software, but the complete sound processing take place in the space of the stage, in relation to the distances and positions of megaphones, microphones, loudspeakers and performers.

The composition is based on the relationships between the two performers and the sound produced through the actions with the feedback systems, performed extemporaneously, but following the indications included in the score. The two performers must shape the sound of the feedback generated by the megaphone systems through their gestures on stage, in terms of amplitude, intonation, length of the feedback, but also in terms of musical expression and theatrical action.

The composition is made according to musical structures created by theatrical actions aimed at representing different roles: a game, a challenge, a plot, a duel. The piece experiments with some possible relationships between voices and feedback systems. The two voices, sometimes in counterpoint, sometimes as soloists, intertwine, merge and blend, or resurface as solos from the electronic texture they have created.

Figure 3. Eleonora Claps (left) and Virginia Guidi (right) perform *Feedback for Two*, by Silvia Lanzalone. Auditorium Parco della Musica, Rome, 2018.

5. CONCLUSIONS

The phenomenon of acoustic feedback is of particular interest for musical composition, as it is directly related to the physical, mechanical, and electrical characteristics of the electroacoustic components involved. The feedback phenomenon is particularly sensitive to micro-variations in its generation system, which can be modified by the interpreter, if integrated with the musical instrument. The unpredictability of the phenomenon given by the feedback must be properly controlled by the insertion of a signal processing algorithm appropriately calibrated on the basis of the characteristics of the specific instrument.

The insertion of a feedback system within the electromechanical system of a musical instrument can give a strong originality to the sound. The timbral "vitality" of

[18] The megaphone model was RCF MG 80

[19] DPA 4060 - Omnidirectional Miniature Microphone

[20] DPA 4088 - Directional Headset Microphone

[21] To preserve the stability of the feedback system I used specific loudspeakers with the following technical specifications: dispersion angle of about 60 degrees, efficiency of about 94 dB 1W / 1m, power of about 130W RMS.

[22] Depending on the size of the hall, the work may have two to four speakers of the same technical characteristics placed at a distance of at least 1.5 meters from each other.

the feedback depends on the variations that can be made on the system by the performer. In fact, the feedback system reacts with great adaptability to external stimuli, matching to significant sound changes as a result of small executive gestures.

In this paper I tried to highlight how, in my work, the phenomenon of acoustic feedback, to be used in the compositional field as an autonomous system, requires an electroacoustic system to generate the positive feedback necessary to self-power the circuit. The implicit instability of the system, distinguished by an exponential and infinitely increasing sound result, requires the adaptive management of a complex set of parameters, some of which are entrusted to musical performance.

Furthermore, the digital signal processing within the feedback loop, allows the use of a very varied but extremely sensitive timbre palette, because it depends on many variables, not only internal, but also external to the system. The performer is no longer a simple player of the instrument but is responsible for the correct sound production and transformation within a complex system. The complex feedback system opposes its action to continuously tend elsewhere, theoretically towards infinity, practically towards its own saturation point. This feature reinforces the performative gesture, renewing its expressiveness.

6. REFERENCES

[1] I. Xenakis, Musique. Architecture, Casterman, Tournai, Belgium, 1976.

[2] G. M. Koenig, Genesi e forma. Origine e sviluppo dell'estetica musicale elettronica, Semar, Rome, Italy, 1995.

[3] P. Inverardi, P Pelliccione, M. Lupone, A. Gabriele, "Ad-Opera: Music-inspired Self-adaptive System", in Computation for Humanity: Information Technology to Advance Society, edited by Justyna Zander, Pieter J Mosterman, CRC Press, 2012, pp. 361-379.

[4] M. Lupone, L. Bianchini, S. Lanzalone, A. Gabriele, M. De Luca, "Struttura, creazione, interazione, evoluzione: la ricerca al CRM per la realizzazione di Forme Immateriali di Michelangelo Lupone", in Proceedings of XXI CIM-Colloquio di Informatica musicale, Cagliari, Italy, 2016, pp.77-84.

[5] M. Palumbi, L. Seno, "Physical modelling by directly solving wave PDE", in Proceedings of ICMC-International Computer Music Conference. Beijing, Republic of China, 1999, pp.325-328.

[6] L. Seno "Modelli fisici e Strumenti musicali", in Acustica Musicale e architettonica, edited by S. Cingolani, R. Spagnolo, UTET, Torino, Italy, 2005, pp 455-494.

[7] S. Lanzalone, "Strumenti aumentati", in Acustica. Fondamenti e applicazioni, a cura di R. Spagnolo, UTET, Torino, Italy, 2015, p.877-888.

[8] E. F. Callery, E. K. Canfield-Dafilou, "Methods for Performing with Feedback in Virtual Acoustics", in Proceedings of the 17th SMC-Sound and Music Computing Conference, Torino, Italy, 2020, pp.138-144.

[9] D. Sanfilippo, A. Valle, "Feedback Systems: An Analytical Framework", Computer Music Journal, The MIT Press, Volume 37, Number 2, Summer 2013, pp.12-27.

[10] M. Lupone, L. Bianchini, S. Lanzalone, A. Gabriele, "Augmented Instruments at CRM - Centro Ricerche Musicali of Rome: Feed-Drum, SkinAct, WindBack and ResoFlute", in Proceedings of ICMC-International Computer Music Conference, New York, USA, 2019.

[11] L. Maiorenzi, L. Seno, "A mathematical model for the holophone, a high directivity acoustical radiator", in atti del XIV CIM-Colloquium on Musical Informatics, Firenze, Italy, 2003, pp.173-176.

[12] M. Lupone, "Musica Elettronica" in Acustica Musicale e architettonica, a cura di S. Cingolani, R. Spagnolo, UTET, Torino, Italy, 2005, pp.526-586.

[13] L. Seno, "Nuova liuteria: caratterizzazione acusto-vibrazionale dei Planofoni®", in Proceedings of Annual Meeting of the Acoustical Engineering Society. Como, Italy, 2005.

[14] S. Lanzalone, "Suoni Scolpiti e Sculture Sonore: alcuni esempi di installazioni d'arte elettroacustica", in Proceedings of XVII CIM-Colloquio di Informatica Musicale, Venezia, Italy, 2008, pp.149-156.

[15] S. Lanzalone, "The 'suspended clarinet' with the 'uncaused sound'. Description of a renewed musical instrument", in Proceedings of 8th International Conference on NIME-New Interfaces for Musical Expression, Casa Paganini-InfoMus Lab, Genova, Italy, 2008, pp.273-276.

[16] S. Lanzalone, "Clavecin électrique. Studio e realizzazione dell'opera", in Proceedings of XIX CIM-Colloquio di Informatica Musicale. Trieste, Italy, 2012, pp. 104-111.

[17] M. Lupone, L. Seno: "Gran Cassa and the adaptive instrument Feed-Drum". Proceedings of Rencontre Musicale Pluridisciplinaires. Lyon, France, 2006, pp 27-36.

[18] W. Cianciusi, L. Seno: "Feed-Drum e SkinAct: l'ultima frontiera dello strumento aumentato". Proceedings of XIX CIM - Colloquio di Informatica Musicale. Trieste, Italy, 2012, pp.72-78.

[19] M. Lupone, S. Lanzalone, L. Seno: "New advancement of the research on the augmented wind instruments: WindBack and ResoFlute". Proceedings of EAW - Electroacoustic Winds 2015. Aveiro, Portugal, 2015, pp.156-163.

[20] L. Russolo, L'Arte dei Rumori, Milano, Italy, 1916.

[21] G. De Simone, Accessi all'opera di Giuseppe Chiari, in "Konsequenz", XIII, n. 13-14 new series, Liguori Editions, Naples, Italy, 2006, pp. 98-108.

GENERATING MUSICAL CONTINUATIONS WITH REPETITION

Sebastian VELEZ DE VILLA (sebastian.velezdevilla@epfl.ch) [1],
Andrew MCLEOD (andrew.mcleod@epfl.ch) (0000-0003-2700-2076) [1], and
Martin ROHRMEIER (martin.rohrmeier@epfl.ch) [1]

[1] *Digital and Cognitive Musicology Lab*, *Digital Humanities Institute*, **École Polytechnique Fédérale de Lausanne**, Switzerland

ABSTRACT

Repetitions play a central role in music, be they repeated themes, harmonies or rhythmic repetitions. They can differ in many ways: vertically, by shifting notes up or down in pitch, time-dilated, by slowing or accelerating them, or just slightly different, for example with ornamentation or removed notes. This work focuses on explicitly generating continuations with patterns given a musical excerpt. We compare two different ways of finding such repetitions, and the found patterns are used to help generate music that is more musically well-formed than a pattern-agnostic approach. Quantitative results show an improvement in performance for both of our algorithms over a pattern-agnostic baseline, with the more sophisticated algorithm exhibiting the most promising results. Qualitatively, it still lacks some creativity, as the model only creates patterns or notes that already exist in the given excerpt, which is particularly an issue for pieces that do not exhibit a large amount of repetition.

1. INTRODUCTION

This paper focuses on monophonic music generation, and in particular on generating music with repetition. Repetition is a fundamental component of musical form [1, 2], from classical music to modern-day pop. Cognitively, it is a major component of a listener's experience with a piece of music, where the prediction and recognition of repeated themes and motives—even (and in some cases in particular) for non-exact repetitions—can lead directly to the enjoyment of the listening experience [3, 4]. Therefore, the inclusion of repetition must form an essential component in any music generation system.

The evaluation of generated music is an extremely subjective and difficult task [5]. Therefore, we concentrate on the first subtask of the MIREX Patterns for Prediction (PP) task [1], whose goal is to develop algorithms that take a short excerpt (a prime) of a piece of music and produce a continuation: some notes that will follow the prime. This allows us to explore music generation with repetition, and has a well-defined quantitative evaluation metric (though we also perform a qualitative evaluation of our results).

The main challenge for our approach is to develop a robust way to find patterns with variation in music. By "robust", we mean an algorithm that can find repeated sequences that would be considered very similar by human listeners: it could be a repeated theme played at a different pitch or tempo, one with pitch or rhythmic variations, or one with added or removed notes. This task is not trivial, even for humans, as the perception of such patterns can be very subjective: even annotators do not have perfect agreement with each other [6].

For a computer, finding such repeated patterns efficiently is even more difficult. It is important to note that hierarchical models of musical form have indeed shown the ability to capture some longer-term, overarching properties of repetition in music, for example in the task of splitting a piece into sections [7]. However, this is quite a different task than detecting the specific local repetitions in which we are interested, and can rely on more global structures such as the harmonic texture of a piece.

A naive approach to finding local repetitions is to compare all subsets of the short excerpt with the whole set, and count how many times each one of them appears in the prime. This solution works for exact matches, but is slow, and doesn't work when the patterns are not exact. It is often the case that themes, or important parts or even sections of the music, are repeated, but those themes commonly undergo variation, transformation or thematic-motivic work throughout a piece. Thus, while the ability to find exact repeats in music is a *necessary* component for our system, it is not a *sufficient* way to ensure that all repeated patterns are found, so we use a more sophisticated approach. In other words, while finding perfect matches is feasible, such patterns are not sufficient for generating music.

Once these repeated patterns are found in our given prime, a continuation must be generated. In recent years, many deep learning approaches have been applied to the task of music generation (e.g., [8–11]). However, such models have a couple of drawbacks for our purposes. For one, we want our model to be adaptable in order to generate music that is strongly informed locally by the given prime. One potential solution for this would be to combine a long-term model (which is trained on a large corpus of music to learn general rules of tonality and form) with a short-term model (which learns the local structure

[1] https://www.music-ir.org/mirex/wiki/2019:Patterns_for_Prediction

Pitch class	C	D	E	F	G
C	0.25	0.5	0.25	0	0
D	0	0	1.0	0	0
E	0.5	0	0	0.5	0
F	0	0	0	0	1.0
G	0	0	1.0	0	0

Table 1: Transition matrix of Figure 1.

specific to a single musical piece). This approach been used by statistical models of music (e.g., [12, 13]); however, it is more difficult to apply this to large networks given the large amounts of training data they are trained on initially and their large parameter count in comparison to such a short prime. Furthermore, given the relatively short length of the continuations required by our chosen task, it is unclear whether such a long-term model would even help. Secondly, with such large black-box-type models, it is difficult to ensure an explicit generation of repeated patterns, as is feasible with simpler alternatives (although some work has been done in this direction, with the goal of enforcing local structural constraints on symbolic music generation based on a template piece [14]).

Therefore, in this work we instead concentrate on statistical approaches to music generation. Specifically, we take existing algorithms for detecting patterns with variations [15, 16], and investigate the results of applying them (with only minor changes) to the task of music generation using a single-order Markov model. Our approaches perform well against the task's simple baseline, described in section 2, but can still lack creativity in some cases, particularly when the prime is not very repetitive.

2. RELATED WORK

In [17], the authors discuss how a computer can generate music using statistics. The basic idea uses Markov chains, i.e., a succession of states, where each state only depends on the previous one. A simple music generation algorithm can be extended from that idea. For example, in [18], a Markov model is trained on a large corpus of pieces and then used to generate a continuation of an input excerpt (similar to our task). However, this model draws from prelearned patterns from its training corpus, rather than patterns taken from the prime directly, as we wish to do.

Consider the simple melody in Figure 1 (the first 4 bars of "Frère Jacques") as an example. Take each pitch class as a state.

To calculate the transition probabilities for each state, a possible algorithm could be formulated in the following way: create a transition matrix, and for each pitch, count how many times a note of that pitch is followed by notes of other (or the same) pitches, and divide by the number of times that pitch appears (disregarding the last note which, by definition, doesn't have a next state). This generates the transition matrix shown in Table 1, which should be read line then column. For example, "D is followed by E with probability 1.0". If the excerpt in Figure 1 is taken as a prime, this table could then be used to generate the continuation by taking the last state (here, "G"), and successively drawing the next note from the distribution in the corresponding row in the table. (A possible output for a 4-note continuation would be "E C E F".) This simple generation algorithm can thus only generate states that appear in the prime sequence (i.e., the input), and can only generate transitions that already exist (e.g., we can never have a C followed by a G, and we can never have a B at all). There is therefore only a very limited notion of creativity or musical structure in this case, only a generation of repetitions of what already exists. To a human listener, most of the outputs from this algorithm sound relatively simple. However, it is fast, and can run in $\mathcal{O}(n+k)$, where n is the length of the sequence, and k is the number of generated states.

In the previous example, the Markov model was used to generate only pitches, but it can also be extended for durations, or pairs of pitches and durations by simply defining the state space differently. In this work, we compare our system against a baseline Markov model trained to generate sequences of (pitch, inter-onset-interval) pairs (i.e., the state space of the model consists of such tuples) [17]. This model is also used as a baseline for the Mirex PP task [2].

2.1 Pattern Detection

In order to improve the basic model of [17], the authors propose a solution based on pattern discovery and pattern inheritance, where a pattern is defined as a set of notes which is repeated, shifted, or time-dilated in the piece. For example, in Figure 1, one pattern would be "C D E C", since it is repeated twice.

The most basic method for pattern discovery is a naive string-based approach which finds only exact repetitions. Many more sophisticated methods exist, none of which is clearly the best in all cases, and they are often designed for very different goals. For example, recent overview of the highly related task of symbolic melodic similarity (in which systems measure the similarity between two musical excerpts), can be found in [19]. In [20], pattern detection was used to perform melodic segmentation by searching for patterns in pattern occurrences. Since they are designed with specific downstream tasks in mind, the above approaches tend to be more complex and less transparent than basic pattern detection algorithms. We therefore draw instead from the basic pattern detection literature, leaving more sophisticated approaches for finding different types of patterns for future work. More thorough discussions on some of these methods (in various contexts) can be found in, e.g., [6, 21–23].

We choose to use a pattern-detection algorithm based on SIA [15] [3], since it was simple to implement, adapt, and investigate for our purposes. We describe SIA and its integration into our generation process in the following section.

[2] https://glitch.com/@tomthecollins/wi-mir-2020-workshop

[3] Although SIA is well-adapted for polyphonic textures, we use it here on monophonic input since it works well, is not slow (given the lengths of our primes), and will allow us to more easily adapt to polyphonic music in future work.

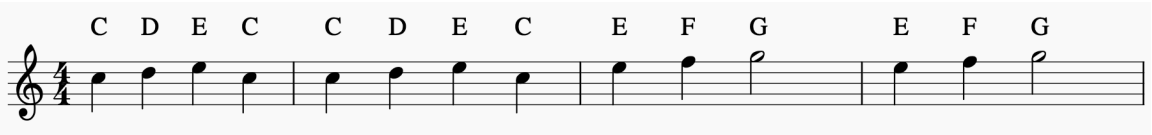


Figure 1. The first 4 bars of "Frère Jacques".

3. DESIGN AND IMPLEMENTATION

3.1 Input format

Our pattern-informed music generation algorithms take as input monophonic sequences of midi notes, which are tuples of pitch (represented as MIDI note number), onset, duration and velocity attributes. Before the detection of patterns and generation, each prime is transformed so that it does not include rests, making both more straightforward. The pitches and onsets remain unchanged, only the durations are updated, so that each note ends when the next note begins. When generating the next state, the starting time of each note is set to the offset time of the previous note.

3.2 Pattern detection

In order to find such patterns, we decided to follow two approaches.

1. A *string-based* approach, which finds exact patterns.
2. A *translation-based* approach, which finds exact patterns, as well as vertically shifted patterns.

Consider the sequence "1 2 3 4 2 3 4 5" consisting of eight states. The string-based solution would only find the pattern "2 3 4", whereas the translation-based algorithm would find the pattern "1 2 3 4", knowing that it is shifted up by one to obtain "2 3 4 5".

3.2.1 String-based pattern recognition

In this case, the notes of the prime are transformed into a list of (pitch, duration) tuples, and the state-space of the Markov model consists of such tuples. The algorithm then uses a string-based pattern matching approach, and finds only exact patterns, defined as a sequence of these (pitch, duration) tuples which appears at least twice in the prime. A pattern detected by this algorithm for Figure 1 would be the first four notes.

3.2.2 Translation-based pattern recognition

As opposed to the string-based approach, this algorithm first transforms the notes of the prime into a list of (pitch, onset) tuples. Then, it uses *translation vectors* to find the maximum translatable pattern: that is, a sequence of notes that can be shifted either vertically (in pitch) or horizontally (in onset time) in the score to find a matching sequence. As opposed to the string-based approach, this method uses onset instead of duration, which is needed to calculate the translation vectors, described below. This algorithm is based on SIA [15], with the difference that we consider only contiguous patterns, which helps for the generation process. We instead leave non-contiguous pattern-based generation for future work.

To make things simple, we will detail this process step by step, again using the simple example of "Frère Jacques" (Figure 1). For each note, we first calculate its translation vector with respect to all following notes in the sequence. Specifically, let n_i and n_j be ith and jth notes of the prime, where $i < j$. Then the translation vector of n_i to n_j is calculated as in Equation 1, where the pitch is represented by its MIDI note number:

$$\begin{pmatrix} n_j.onset - n_i.onset \\ n_j.pitch - n_i.pitch \end{pmatrix} \tag{1}$$

Considering the first two bars of "Frère Jacques," This results in the table shown in Table 2. (Durations are measured in whole notes here, but any other representation would be equivalent, as long as it is consistent throughout a prime.) Here, the table should be read starting with columns and then rows (e.g., the first note (0.0, 72) can be transformed into the second note (0.25, 74) by adding the translation vector (0.25, 2). Then for each translation vector that appears at least twice in the table (signified by colors), we create a sequence of those notes which have this vector in their column. Sorting the translation vectors by the length of the resulting sequence of notes results in:

- $\begin{pmatrix} 0.25 \\ 2 \end{pmatrix}$: $\begin{pmatrix} 0.0 \\ 72 \end{pmatrix}$, $\begin{pmatrix} 0.25 \\ 74 \end{pmatrix}$, $\begin{pmatrix} 1.0 \\ 72 \end{pmatrix}$ and $\begin{pmatrix} 1.25 \\ 74 \end{pmatrix}$.
- $\begin{pmatrix} 1.0 \\ 0 \end{pmatrix}$: $\begin{pmatrix} 0.0 \\ 72 \end{pmatrix}$, $\begin{pmatrix} 0.25 \\ 74 \end{pmatrix}$, $\begin{pmatrix} 0.5 \\ 76 \end{pmatrix}$ and $\begin{pmatrix} 0.75 \\ 72 \end{pmatrix}$.
- $\begin{pmatrix} 0.5 \\ 4 \end{pmatrix}$: $\begin{pmatrix} 0.0 \\ 72 \end{pmatrix}$ and $\begin{pmatrix} 1.0 \\ 72 \end{pmatrix}$.
- ...
- $\begin{pmatrix} 1.25 \\ 2 \end{pmatrix}$: $\begin{pmatrix} 0.0 \\ 72 \end{pmatrix}$ and $\begin{pmatrix} 0.25 \\ 74 \end{pmatrix}$.

Each of these sequences is a potential repeated pattern. They are filtered to keep only those sequences which contain only contiguous notes, like the second one in the enumeration above (shown in purple in the table), and unlike the first one (in red). This filtering eliminates any "split" patterns whose beginning and end each repeats, but whose middle changes. We also remove any patterns whose repetition contains any notes from its original occurrence. Starting from the top of this sorted list, a pattern is valid when no notes of that pattern have appeared in a previous valid pattern.

(Onset, pitch)	(0.0, 72)	(0.25, 74)	(0.5, 76)	(0.75,72)	(1.0, 72)	(1.25, 74)	(1.5, 76)	(1.75, 72)
(0.0, 72)	-	-	-	-	-	-	-	-
(0.25, 74)	(0.25, 2)	-	-	-	-	-	-	-
(0.5, 76)	(0.5, 4)	(0.25, 2)	-	-	-	-	-	-
(0.75, 72)	(0.75, 0)	(0.5, -2)	(0.25, -4)	-	-	-	-	-
(1.0, 72)	(1.0, 0)	(0.75, -2)	(0.5, -4)	(0.25, 0)	-	-	-	-
(1.25, 74)	(1.25, 2)	(1.0, 0)	(0.75, -2)	(0.5, 2)	(0.25, 2)	-	-	-
(1.5, 76)	(1.5, 4)	(1.25, 2)	(1.0, 0)	(0.75, 4)	(0.5, 4)	(0.25, 2)	-	-
(1.75, 72)	(1.75, 0)	(1.5, -2)	(1.25, -4)	(1.0, 0)	(0.75, 0)	(0.5, -2)	(0.25, -4)	-

Table 2: Translation vectors of the first two bars of Figure 1. Colors signify identical translation vectors (potential patterns), and uncoloured, non-empty cells are unique. The pitches are indicated by their MIDI note number.

So, in this example, the first four notes (corresponding to the columns of the purple translation vectors in the table) can be shifted by four beats to obtain the last four notes (corresponding to the rows of the purple vectors in the table), and are saved as a valid pattern.

3.3 Special cases

For both the string-based and translation-based algorithms, when a note is not part of any larger pattern, it is considered as a pattern itself, making it easier for the generation. In the case where the last pattern of the prime is unique, the probabilities for the next pattern are set to the unigram probability of each pattern appearing in the prime, regardless of position.

3.4 Generation

Once the patterns are found, both algorithms work in the same way: they transform the sequence of notes into a sequence of patterns, and apply a first-order Markov model on this transformed sequence to generate, not a continuation of notes, but rather a continuation of patterns. Then, we translate this sequence of patterns back into a sequence of midi notes: a list of tuples with pitch, onset, duration and velocity (which we always set to 100).

3.5 Smoothing

These algorithms as presented can only produce transitions that already exist in the given prime. In order to inject additional creativity into the generation, we apply a form of additive smoothing as follows. First, we produce the transition matrix over the patterns found in the prime (as shown in Table 1 for pitches). Then, each probability is multiplied by some number $\alpha < 1$, thus removing some probability mass from the transitions found in the prime. We then distribute the remaining $1 - \alpha$ probability mass among the states, proportional to the normalised histogram (the unigram probability of each state; shown in Table 3 for "Frère Jacques"). In our experiments, we set α to 0.9.

This approach ensures that states that often appear in the prime also appear more often than others in the generation, but would still create transitions that don't exist in the prime, resulting in some "creativity". With this smoothing, the transition matrix shown in Table 1 is changed as in Table 4.

C	D	E	F	G
0.2857	0.1429	0.2857	0.1429	0.1429

Table 3: The normalised histogram (unigram probabilities) of the pitches of Figure 1.

Pitch class	C	D	E	F	G
C	0.25357	0.46429	0.25357	0.01429	0.01429
D	0.02857	0.01429	0.92857	0.01429	0.01429
E	0.47857	0.01429	0.02857	0.46429	0.01429
F	0.02857	0.01429	0.02857	0.01429	0.91429
G	0.02857	0.01429	0.92857	0.01429	0.01429

Table 4: Transition matrix of Figure 1 with additive smoothing ($\alpha = 0.9$).

4. EVALUATION

4.1 Baseline Methods

We compare against two different baseline methods, neither of which uses any sort of pattern detection or explicit repetition generation. The first, *Baseline*, is based on the model described in [17], and is also used as a baseline in the MIREX PP task. It is a first-order Markov model which is trained on and outputs (inter-onset-interval, pitch) tuples. The second, *Simple*, outputs the combination of two first-order Markov models: one which outputs the pitch of each note and a second which outputs the inter-onset interval (IOI) for each.

4.2 Metrics

For a quantitative evaluation, we apply the two metrics also used in the MIREX PP task: cardinality score and pitch score. [4]

4.2.1 Cardinality score

The cardinality score (CS) is defined as:

$$\mathrm{CS}(\mathbf{P}, \mathbf{Q}) = \max_{t \in \mathbf{T}} |\{q \,\forall\, q \in \mathbf{Q} \mid (q + t) \in \mathbf{P}\}| \quad (2)$$

Here, **P** and **Q** sets of (onset, pitch) tuples for the true and generated continuations, respectively, and **T** is the set of all possible translation vectors that make a note from **Q**

[4] The code used for evaluation is available at `https://github.com/BeritJanssen/PatternsForPrediction`.

overlap a note from **P**, formally defined as:

$$\mathbf{T} = \{p - q \,\forall\, p \in \mathbf{P}, q \in \mathbf{Q}\} \tag{3}$$

In other words, a higher score reflects how similar two continuation are in their general shape, ignoring shifts in time and pitch. Using this score, recall and precision can be calculated as in Equations 4 and 5 (note that we subtract 1 from each numerator and denominator since at least one note is guaranteed to overlap), and F1 is calculated as their harmonic mean as usual. Intuitively, recall is the proportion of the continuation which has been correctly generated, and precision is the proportion of the generation that matches the continuation.

$$Recall = \frac{\text{CS}(\mathbf{P}, \mathbf{Q}) - 1}{|\mathbf{P}| - 1} \tag{4}$$

$$Precision = \frac{\text{CS}(\mathbf{P}, \mathbf{Q}) - 1}{|\mathbf{Q}| - 1} \tag{5}$$

The cardinality scores are also plotted as a function of beats after the prime's final onset position, where each CS value considers only notes from the true continuation up to that position. So, if a generation contains a contour that matches the beginning of the true continuation, but not the end (as might be expected), that generation's CS will be higher for smaller onset values (after some possible initial values of 0 corresponding to positions before the 2nd generated note). The CS at onset 10 corresponds to the overall cardinality score with respect to 10 beats after the final note from the prime, and thus represents the best indicator of a generation's over quality in this regard.

4.2.2 Pitch score

One issue with the cardinality score is that it is pitch invariant: if the generated continuation is equal to the true continuation but shifted vertically, it would get a perfect score of 1, since the whole sequence of notes can be translated into the true continuation with only one translation vector. Pitch score is used to solve this problem. It is the magnitude of the overlap between normalised histograms of the true and generated continuations (an example of a normalised histogram is given in Table 3). This process is repeated disregarding octaves (i.e., taking each pitch modulo 12).

4.3 Data

The dataset used for evaluation is composed of two parts: the prime, from which a continuation will be generated, and the true continuation, which will be compared with the outputs of the different generation systems. Specifically, we use the large monophonic dataset prepared for the MIREX PP task, which consists of excerpts taken from the Lakh MIDI dataset [24].

4.4 Quantitative Results

Each system's CS with respect to onset position are plotted in Figure 2. We can first observe that the simple first-order Markov model performs poorly, which is somewhat expected. The next best model appears to be the baseline, and both of our systems with pattern recognition achieve an improvement in precision, recall, and F1, with the translation-based system performing the best. Our systems see the greatest improvement in terms of recall, which suggests that they output more notes of the correct general shape. Our systems also avoid the sharp decrease in performance of the baseline system over the first few beats, instead seeing a slow decrease across the duration of the continuation. This makes sense conceptually, because instead of generating one *note* at a time, our systems generate one *pattern* at a time. Thus, they are typically fewer steps along in the generation process after any given duration (each step holds a potential for the generation to go off track).

Model	**mean**	**median**	**std**
Baseline [17]	0.542	0.555	0.211
Simple	0.544	0.556	0.206
String-based	0.553	0.565	0.220
Translation-based	0.574	0.588	0.218

Table 5: Pitch scores.

Model	**mean**	**median**	**std**
Baseline [17]	0.616	0.635	0.188
Simple	0.618	0.633	0.185
String-based	0.627	0.650	0.196
Translation-based	0.647	0.670	0.192

Table 6: Modulo 12 pitch scores.

Table 5 shows each system's pitch scores (both with and without octave equivalence) in violin plots, and the exact values are given in Tables 5 and 6. We can observe that the systems are quite similar. However, the translation-based system again achieves the best results, though only marginally in this case. The scores for all systems are only slightly above 0.5, which shows that the majority of generated notes are of the correct pitch, but there are still many incorrect notes from this perspective. Of these errors, fewer than 10% are simple octave errors (this can be measured by the difference between the values in Tables 5 and 6).

4.5 Examples

For a more in-depth comparison of the performance of our approaches to pattern-based generation, we now present an in-depth analysis of the translation-based, string-based, and simple (no pattern) outputs for two example primes. In all figures in this section, the string-based and translation-based patterns are annotated with red and green brackets, respectively.

The first example is a prime from our test dataset, shown in Figure 4. From the red and green annotations, it is clear that the translation-based pattern detection algorithm has found longer patterns on average than the string-based one. In particular, there are many more single-note "patterns" for the string-based algorithm. The true continuation, as well as the continuation generated by each system, are

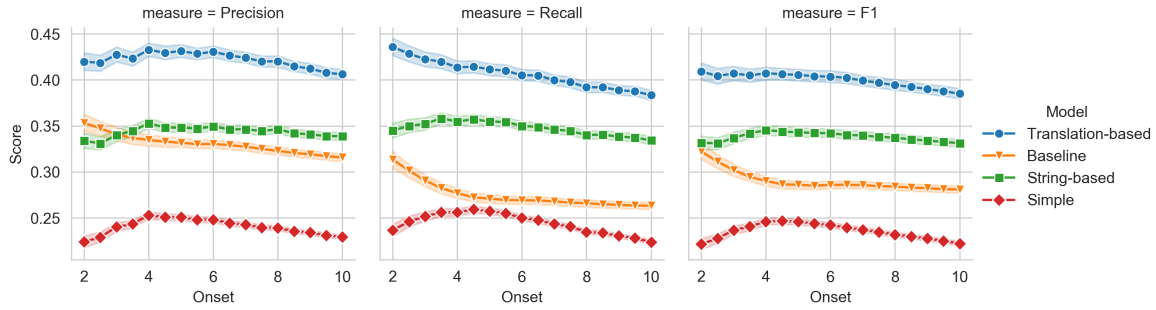


Figure 2. Cardinality scores for our proposed systems (Transition-based and String-based) as well as the simple Markov model and the baseline. The x-axis represents onset position in the true continuation, measured in quarter notes.

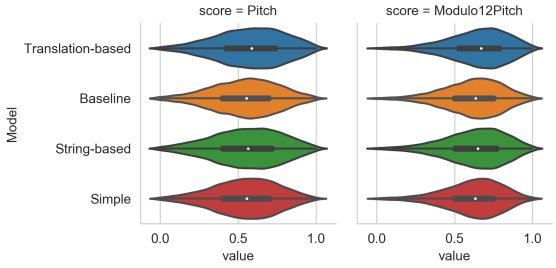


Figure 3. Pitch scores for our three models and the baseline.

shown in Figure 6 (note that the continuations are sampled from distributions, so will change each time). We can observe that the generation without pattern recognition is the least similar to the true continuation. The string-based generation is better: it has the correct rhythm, and even nearly the correct pitch contour for the first half (though notes are shifted up by both minor and major thirds). However, the translation-based generation matches the true continuation exactly.

Clearly, the translation-based approach performs quite well when repetition is present in the prime. However, that is also its drawback: its generations heavily depend on the structure of the prime: if the prime is not repetitive, it will not be able to rely on such pattern generations to produce, and the generation could be poor.

The example prime shown in Figure 5 (the "Castle in the sky" theme, composed by Joe Hisaishi) is one such case. Notice how many small patterns are found by both methods, and how short each one is. The generations (shown in Figure 7) reflect this: only very short patterns can be generated, and none of the generations match the true continuation very well. The string-based generation is slightly better in terms of rhythm, at least matching the dotted-half note, quarter note rhythm in bars two and four, as well as the position of the dotted-quarter notes in bars one and three. In terms of pitch, none of the generations perform very well, although they produce a reasonable set of notes.

So, it can be seen that the translation-based method produces the most accurate generations for repetitive primes, but falls back to around the performance of the less sophisticated systems for primes without much repetition.

5. CONCLUSION

In this work, we presented two novel systems for generating music with explicit repetition, based on a given prime. The systems generally work by first detecting repeated patterns in the prime, and using these to inform the generation process with a simple Markov model. We show that a more flexible, translation-based pattern detection algorithm is able to capture more sophisticated forms of repetition, which improves its generations. Overall, this pattern-based approach works well when the prime is somewhat repetitive; however, it can struggle otherwise.

This reliance on patterns is another potential drawback of our system in that it has no mechanism to make small changes to the found patterns. The creativity involved in making small changes to repeated patterns throughout a piece of music is very important to such repetition, and our system currently lacks this ability. Future work could try to improve this in two ways. First, during pattern detection, the algorithm could be adapted to find such patterns with variations, explicitly noting the types of variations seen in the prime. Then, during generation, the model could explicitly add some of these or other variations into the generated patterns. This would allow the system to produce more creative generations, while still ensuring that it has some repetitive structure. The evaluation of music generation is a very difficult problem, and in future work, we could also include a subjective evaluation involving a group of experts, especially when using primes that do not show repeated patterns (since these generations can be the most varied).

In this work, we concentrated on monophonic music, but similar algorithms for polyphonic music can also be developed in a few ways. The simplest is to apply a voice separation model (e.g., [25]) as a preprocessing step, then producing one generation per voice. Another option is to enlarge the state-space of the Markov model ton include combinations of notes, although this adds a significant amount of complication to the process.

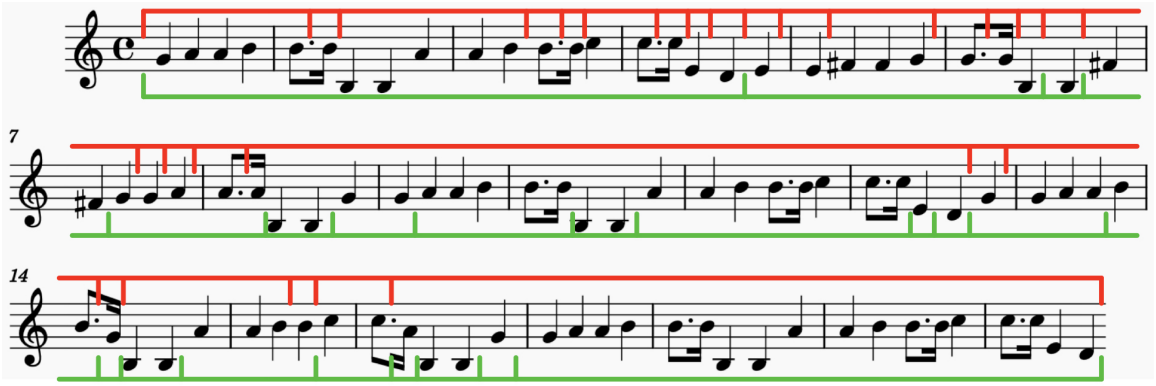


Figure 4. MIDI sample taken from the MIREX 2019: Patterns for prediction dataset. Red and green brackets show the patterns found by the string-based and translation-based algorithms, respectively.

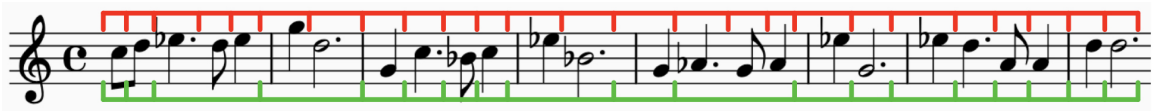
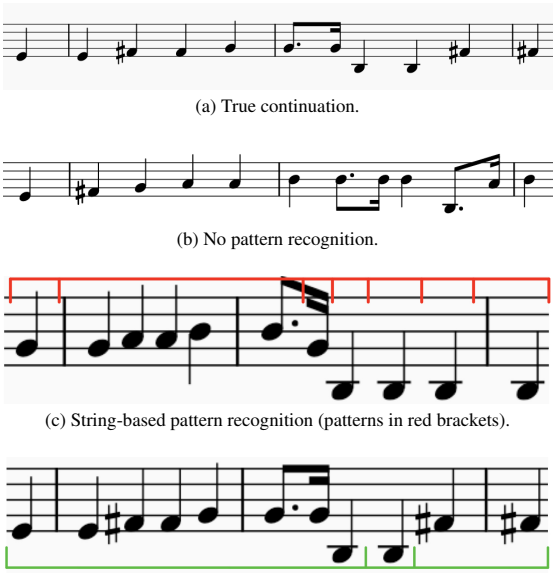


Figure 5. "Castle in the sky" theme, composed by Joe Hisaishi. In red, the patterns found by the string-based approach, in green, the ones found by the translation-based algorithm.



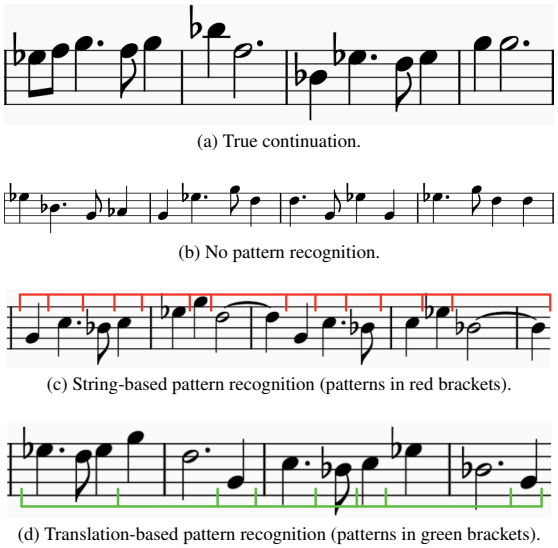
(a) True continuation.

(b) No pattern recognition.

(c) String-based pattern recognition (patterns in red brackets).

(d) Translation-based pattern recognition (patterns in green brackets).

Figure 6. Generated and true continuations of the prime shown in Figure 4. Exact onset timing has been quantized to the nearest 16th note.



(a) True continuation.

(b) No pattern recognition.

(c) String-based pattern recognition (patterns in red brackets).

(d) Translation-based pattern recognition (patterns in green brackets).

Figure 7. Generated and true continuations of the prime shown in Figure 5. Exact onset timing has been quantized to the nearest 16th note.

The code for this work is available online [5] .

Acknowledgments

Research supported through the Swiss National Science Foundation within the project "Distant Listening – The Development of Harmony over Three Centuries (1700–2000)" (Grant no. 182811).

6. REFERENCES

[1] A. Ockelford, *Repetition in music: Theoretical and metatheoretical perspectives*. Routledge, 2017.

[2] W. E. Caplin, *Classical form: A theory of formal functions for the instrumental music of Haydn, Mozart, and Beethoven*. Oxford University Press, 2001.

[3] D. Huron, *Sweet anticipation: Music and the psychology of expectation*. MIT press, 2008.

[4] E. H. Margulis, *On repeat: How music plays the mind*. Oxford University Press, 2014.

[5] B. L. Sturm and O. Ben-Tal, "Taking the models back to music practice: Evaluating generative transcription models built using deep learning," *Journal of Creative Music Systems*, vol. 2, no. 1, 2017.

[6] B. Janssen, P. van Kranenburg, and A. Volk, "Finding occurrences of melodic segments in folk songs employing symbolic similarity measures," *Journal of New Music Research*, vol. 46, no. 2, pp. 118–134, 2017.

[7] G. Shibata, R. Nishikimi, E. Nakamura, and K. Yoshii, "Statistical music structure analysis based on a homogeneity-, repetitiveness-, and regularity-aware hierarchical hidden semi-markov model." in *ISMIR*, 2019, pp. 268–275.

[8] B. Sturm, J. F. Santos, and I. Korshunova, "Folk music style modelling by recurrent neural networks with long short term memory units," in *16th International Society for Music Information Retrieval Conference*, 2015.

[9] J.-P. Briot, G. Hadjeres, and F.-D. Pachet, "Deep learning techniques for music generation – a survey," 2019.

[10] L.-C. Yang, S.-Y. Chou, and Y. Yang, "Midinet: A convolutional generative adversarial network for symbolic-domain music generation," in *International Society for Music Information Retrieval Conference*, 2017.

[11] Z. Wang, Y. Zhang, Y. Zhang, J. Jiang, R. Yang, J. Zhao, and G. Xia, "Pianotree VAE: Structured representation learning for polyphonic music," in *International Society for Music Information Retrieval Conference*, 2020.

[12] D. Conklin, "Prediction and entropy of music," Master's thesis, The University of Calgary, 1992.

[13] M. T. Pearce, "The construction and evaluation of statistical models of melodic structure in music perception and composition," Ph.D. dissertation, City University London, 2005.

[14] S. Lattner, M. Grachten, and G. Widmer, "Imposing higher-level structure in polyphonic music generation using convolutional restricted boltzmann machines and constraints," *Journal of Creative Music Systems*, vol. 2, pp. 1–31, 2018.

[15] D. Meredith, K. Lemström, and G. A. Wiggins, "Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music," *Journal of New Music Research*, vol. 31, no. 4, pp. 321–345, 2002.

[16] P.-Y. Rolland, "Discovering patterns in musical sequences," *Journal of New Music Research*, vol. 28, no. 4, pp. 334–350, 1999.

[17] T. Collins, R. Laney, A. Willis, and P. H. Garthwaite, "Chopin, mazurkas and markov," *Significance*, vol. 8, no. 4, pp. 154–159, 2011.

[18] F. Pachet, "The continuator: Musical interaction with style," *Journal of New Music Research*, vol. 32, no. 3, pp. 333–341, 2003.

[19] V. Velardo, M. Vallati, and S. Jan, "Symbolic melodic similarity: State of the art and future challenges," *Computer Music Journal*, vol. 40, no. 2, pp. 70–83, 06 2016.

[20] E. Cambouropoulos, "Musical parallelism and melodic segmentation:: A computational approach," *Music Perception*, vol. 23, no. 3, pp. 249–268, 2006.

[21] D. Meredith, "Cosiatec and siateccompress: Pattern discovery by geometric compression," in *International society for music information retrieval conference*. International Society for Music Information Retrieval, 2013.

[22] I. Y. Ren, H. V. Koops, A. Volk, and W. Swierstra, "In search of the consensus among musical pattern discovery algorithms," in *Proceedings of the 18th International Society for Music Information Retrieval Conference*. ISMIR press, 2017, pp. 671–678.

[23] P. Boot, A. Volk, and W. B. de Haas, "Evaluating the role of repeated patterns in folk song classification and compression," *Journal of New Music Research*, vol. 45, no. 3, pp. 223–238, 2016.

[24] C. Raffel, "Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching," Ph.D. dissertation, Columbia University, 2016.

[25] A. McLeod and M. Steedman, "HMM-based voice separation of MIDI performance," *Journal of New Music Research*, vol. 45, no. 1, pp. 17–26, 2016.

[5] https://github.com/v1ruso/BachelorProject

A MODULAR TOOL FOR AUTOMATIC SOUNDPAINTING QUERY RECOGNITION AND MUSIC GENERATION IN MAX/MSP

Arthur PARMENTIER (arthur.parmentier@epfl.ch) [1], **Ken DÉGUERNEL** (ken.deguernel@epfl.ch) [1], and **Constance FREI** (constance.frei@unil.ch) [2]

[1] **Ecole Polytechnique Fédérale de Lausanne**, Switzerland
[2] **Université de Lausanne**, Switzerland

ABSTRACT

This paper presents a modular software designed for automatic recognition of Soundpainting query. Soundpainting is a musical practice and a sign language used for real-time composition with an orchestra. A series of signs is gestured by the soundpainter creating a "sentence" describing a musical idea that they want the orchestra to perform. We propose an open-source tool, for Max/MSP, able to perform every task for a computer to be part of a Soundpainting scene: motion tracking, gesture recognition, query parsing and music generation. This tool is created in a modular way so that it can be easily modified to fit the needs of the user, for instance, changing the type of inputs for the motion tracking or adapting the Soundpainting grammar. We describe the global architecture, the different components of this tool, and the currently implemented methods for each of these components. We then show examples of use for this tool, from the learning of a new sign to a performance with several virtual instruments.

1. INTRODUCTION

Soundpainting (SP) is a sign language developed in 1974 by the New York composer and saxophonist Walter Thompson for real-time composition with his orchestra [1]. Although the language was originally used for composing with musicians, it has extended to multiple artistic disciplines such as dance, theatre or visual arts and is now used worldwide by a variety of artists in diverse contexts and configurations. SP is not originally a language designed for working with electronic devices and computers. It is often reported that their use in SP is made difficult by the high reactivity requested by the soundpainter to the set of performers that forms the orchestra. However, digital tools have been used since the second half of the 20th century in new forms of compositional processes and aesthetics of music [2–4]. Modern methods of Human-Machine Interaction for learning and performing in real-time [5, 6] enables the exploration of new artistic materials with new dynamics of creativity [7, 8].

Automatic gesture recognition is an important topic of Human-Machine Interaction aiming at interpreting human gestures using cameras and/or motion sensors, providing 2D or 3D information. Machine learning techniques are used in order to recognise and analyse the gestures, usually focusing either on full-body gesture recognition or hand gesture recognition. Automatic gesture recognition is used for many applications, such as medical diagnosis [9], surgery analysis [10, 11], emotion recognition [12], sign language recognition [13], etc.

Automatic gesture recognition has also been used in many applications for computer music [14]. For instance, Fernández et al. [15] use Kinect camera, controller gloves and sound mapping for audiovisual performances inspired by percussionists movement; Dalmazzo et al. [16] use a Myo armband to detect arms and fingers movement for a pedagogical tool checking the bowing and fingering of a violinist; Cavdir et al. [17] present an interface with hand gesture recognition and haptic feedback with movement inspired by sign language to create musical performances that can be experienced both by people who are hard of hearing and by those who are not; Chandran et al. [18] uses a camera and openCV to detect facial expressions to control virtual instruments; Van Nort et al. [19, 20] proposes a gesture database, in order to analyse and map sonic and kinetic actions happening during performances according to their gestural meaning, etc.

In the last few years, automatic gesture recognition for SP started gathering some steam. Pellegrini et al. [21] proposed a proof of concept with the recognition of a few SP gestures using Kinect input classified with Hidden Markov Models. However, at the time, they only discussed the possibility of SP annotations and other use cases of the recognition system theoretically without proposing a concrete prototype for it. More recently, Gómez Jáuregui et al. [22] used SP gesture recognition for the generation of electronic music. They use a Kinect and decision tree classifier to recognise 9 different gestures. However, the gestures are considered individually and not as part of a syntactic phrase. SP gesture recognition has also been used outside of the scope of music in order to control a swarm of drones [23].

In this paper, we propose a new tool for Soundpainting Query Recognition (SPQR) with two main aspects in mind. First, the software we propose is complete, ie. it performs every necessary task for SPQR (cf. Figure 2) motion tracking, real-time classification of gesture, query parsing and music generation, and can be used by anyone with a com-

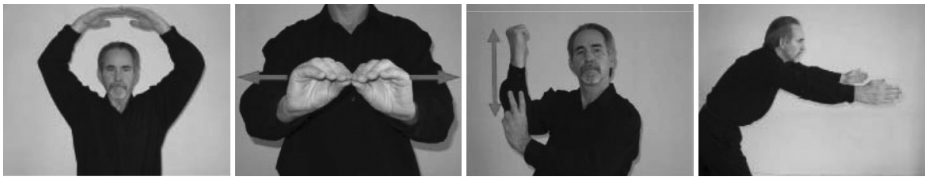


Figure 1. Example of a Soundpainting query using the Who? What? How? When? syntax (illustration from [1]).

puter and a webcam. Second, the software is built in a modular way: every component is independent, enabling an advanced user to easily change them, for instance modifying the motion censors or replacing the classification method.

The rest of the paper is structured as follows. In section 2, we introduce and discuss the key elements of SP related to our project. Then, in section 3, we describe the architecture and the different components of our SPQR software. Then in section 4, we discuss some aspects of performance and provide demonstrations for the use of the software. Finally, in section 5 we discuss the use of this tool in the SP sphere and propose some perspectives for the future of this project.

2. ELEMENTS OF SOUNDPAINTING

Soundpainting is a sign language used by a (or in some cases, several) composer, called soundpainter, as a real-time communication system with performers of an orchestra. The language is organised in "modes" of interaction between the soundpainters and the performers, each one having its own grammar and dictionary of signs. The most common mode ("default mode") uses a syntax that is commonly simplified as "Who" –indicating which performers should respond to the request–, "What" –what content should they perform–, "How" –the shape of the content– and "When" –when the request should be executed–. Fig. 1 shows an example of a SP phrase using this syntax. In response, the performers do not sign but rather provide artistic contents that build up the composition. The soundpainter can then sign in reaction or not to the performers' responses shaping the artistic material while it is being played to the audience. In the rest of the article, we call "query" a SP phrase. This is a bit of a play on word between the computer science definition that the software has to deal with, and the human language definition, since in the philosophy of SP, the soundpainter only requests things from performers without any guarantee on the performer's responses. *"The Soundpainter composes with what happens in the moment, whether expected or not."*

Most of the basic signs of SP involve only a combination of dynamic gesture, e.g. arms and hands movements and of static postures made by the soundpainter, facing the performers. For instance, in the SP workbook [1] the sign for "Whole-group" (see Fig. 1) is described as the following posture: *"Hold both arms over your head creating a circle with fingertips barely touching"*, and the sign for long-tone, is described as the following gesture: *"Holding your hands a little out in front of your body, pinch the thumb and index finger of both hand together and pull them apart along a horizontal plane"*. However, it is to note that the time and space span for each gesture and posture may vary.

Among the most common SP signs are the "multi-disciplinary" signs. These signs such as "minimalism" or "long tone" can be interpreted in several disciplines (music, dance, acting, visual arts...). Even though the concept they refer to usually come from one discipline only, their interpretation in SP has been extended to the other ones by creating relevant analogies for each one. It results in a rich and powerful language that stimulates the creativity of the composer and performers using it.

The full grammar of SP is not yet fully described and involves many other syntactic categories that are learned during practice, therefore, for the rest of this article, we are only considering the "default" mode for which the syntax is established and described in details in [1].

3. SOFTWARE ARCHITECTURE

SPQR is built with several independent layers, emulating the different structural levels of SP, from the creation of signs to the parsing of its grammar and music generation. We have identified five layers:

- **input management**, computing features from different input systems (webcam, gloves, etc.),
- **sign and dictionary managemen**t, defining and storing the SP gestures and postures,
- **real-time classification,** performing SP sign recognition,
- **parsing and request-forming automata**, creating a query out of a series of signs,
- **orchestra simulation**, performing music according to the query.

An illustration of these components and their interactions is given in Figure 2.

Each layer was conceived as a specific function that the user should easily be able to identify and interpret. Inside each layer are different processes and objects that the user interacts superficially with from the interface of the program.

At the interface level, all layers are implemented inside Max/MSP. The user can see the whole patcher in

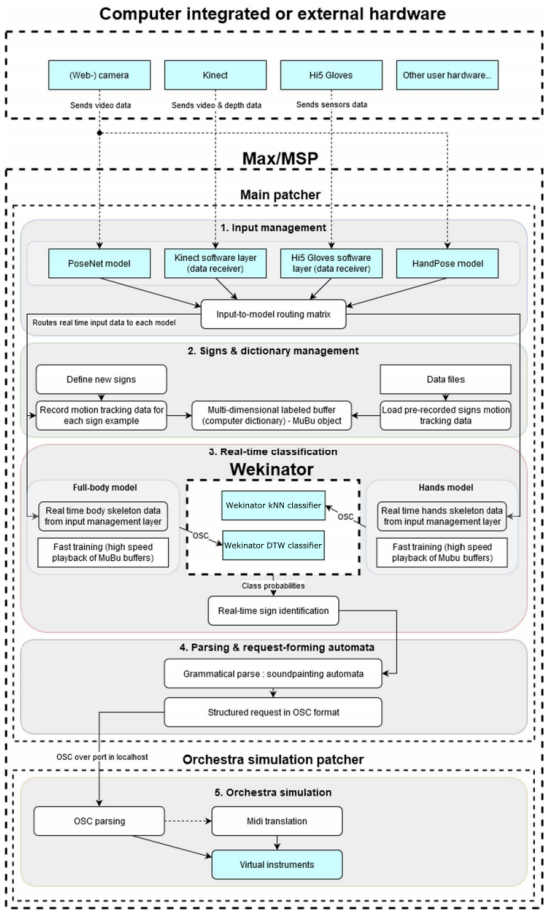


Figure 2. Schematic overview of the different modules and their interactions in SPQR. Blue components come integrally or partially from other works while white components have been designed during this project.

the main window and is also able to access specific functionalities of each layer by using tabs. At the processing level, Max/MSP itself has three different threads that it uses for processing the data passing through its compiled objects. For these threads, Max guarantees the synchronicity/ordering of events. However, Max also interprets Node.js code that is processed in external threads asynchronously to Max internal threads.

In this part, we describe the different components implemented in the current version of this software, but each module could be easily replaced by an experienced user.

3.1 Input management

The role of the motion tracking layer is to compute a set of motion features from the movement of the user. There exist several motion tracking systems with different technologies that can be used to model the human body from the position of characteristic points. These points can then be transformed into features of a classifier to identify gestural signs. In SP, some body parts such as the hands are used much more frequently to sign than others, therefore they require more precise tracking than the latter to classify amongst the signs. However, all motion tracking systems available have a finite range of operation, i.e. they can only track motion at a certain scale (just like the human cognition system). We integrated two models to respectively track signs from the full-body and from the hands.

For these scales, we use respectively PoseNet [24] –an open-source computer-vision model that can be used to estimate the pose of a person in an image or video by estimating where key body joints are in 2D space– and HandPose [25] –it's equivalent for modelling the hand–. For PoseNet, the available features are the position coordinate for ears, eyes, shoulder, elbows, wrists, hips, knees and ankles with the nose considered as a fixed point. For HandPose, the available features are the position coordinate of every finger tip and knuckles with the centre of the palm considered as a fixed point. For our experiments, we only used the positions of elbows and wrists for PoseNet, and the positions of every fingertip as well as the second knuckle for the index and middles fingers.

Their performances on modern CPUs and GPUs allow them to run in real-time using a webcam or alternative low-latency video input devices, making them usable by anyone without needing special hardware. We use the TensorFlow version of these tools which are ported into Max/MSP using a Node.js server. Therefore, the patch provides a simple user interface (cf. Figure 3). PoseNet and HandPose allow the user to choose different models and internal parameters that will affect its performance, such as the architecture of the model (MobileNet or ResNet), the input resolution of the video, the size or the depths of the model, etc. With these settings, the user can adapt the model to its hardware easily to get the best performance.

The different inputs can be attributed to the different models using the input manager. We opted for a modular approach with a design that allows the user to add its own inputs and models in from the Max patcher. Because inputs have different data rates and dimensions, it is in general not possible to route more than one input to a single model. If two inputs are compatible (typically with similar data rates) the user simply can create a new input and merge the two original ones.

3.2 Signs and dictionary management

Once the inputs and models are defined in the program, the user can start recording signs and building its sign dictionaries for each model. Users can either create their own signs or use pre-recorded signs, that would have been created or recorded by other users. A sign can be defined by two properties: its name or its category, in analogy with the syntactic model of SP (identifier, content, etc.) These properties are sufficient to allow the program to parse the sign, i.e. to construct a meaningful request from the temporal flow of signs.

In order for the sign to effectively be identified, two steps are required after the sign has been defined: record training examples and program the virtual instrument itself to interpret the sign. While the recording of training exam-

Figure 3. User interface for PoseNet in the Max/MSP patch. The left part shows the input from a webcam with the PoseNet key body joints. The right part shows the different coordinate for each key body joints. The user can select here which features to take into consideration in the system.

ples is an automated process that simply involves pushing one button, the programming of the virtual instrument or device that the sign should control is outside the scope of the program.

The user can choose to either define one sign at a time and record one or several training examples for it, then saving the training data and adding another sign, or directly define a list of signs and recording all of them in the same session. Each recording takes place in a different buffer of the Multiple Buffer (MuBu) objects (one Mubu object per model) [26] and each active input data is saved into a different track. The recordings can then be saved to build a dictionary of MuBus corresponding to different signs. It is also possible to load pre-recorded signs by dragging and dropping data files in the dedicated zone.

3.3 Real-time classification

For the system to be able to "recognise" the signs, we decided to focus on lightweight, interpretable models that can be trained fast and identify the signs that are performed in real-time. In our case, the identification is a simple classification process, in which we ask the classifier to predict the "class" of the motion sequence performed among a set of classes that have been previously learned by the model –the SP signs.

After a few experiments with the gesture follower from MuBu based on Hierarchical Hidden Markov Models, we decided to use the external software Wekinator [27] performing better by offering a very efficient Dynamic Time Warping implementation based on the FastDTW library [28] with additional improvements for real-time performance and several internal parameters for its DTW model. Wekinator is integrated into Max through communication with the Open Sound Control (OSC) protocol. Although the user must launch Wekinator separately and perform basic operations on its GUI, the most important parts of Wekinator can be controlled remotely via OSC, allowing Max to automatise certain operations, such as its training process.

Once the model is running, Max receives in real-time the set of DTW distances from the real-time sequence to each recorded sign sequence. By finding the minimal value in that set and comparing this value to a confidence threshold, we can identify when a sign is being performed in real-time.

3.4 Parsing and request-forming automata

From the models introduced in the previous section, we can recognise individual signs, forming a sequence in time, just like words form a phrase in oral languages. The next step is therefore to implement the grammar of SP with a parsing mechanism that would then allow us to create requests or commands to each device that acts as an individual performer in the system. To that purpose, we implemented an automaton inside Max using Node.js. A visualisation of the automaton is provided inside the patch for a clear description of the SP grammar and also for more experienced users to have direct feedback on their grammatical implementation, for instance when adapting the grammar to their own purpose (cf Figure 4).

The automaton that we implemented corresponds to the grammar for the "default mode" of SP explained in section 2 (the "Who–What–How–When" syntax). The automaton allows us to represent such a sequence and understand each sign as a particular function in the query. It also recognises "wrong" signs and allows for feedback for the user about the correctness of their query. Details about the grammar of SP and its implementation can be found in [29].

Once parsed, the query is structured by collecting each sign during the state transitions and assembling them into several hierarchical objects. The query is then converted in OSC format that can be used to communicate with a virtual performer.

3.5 Orchestra simulation

From the OSC commands created by the automaton, there is an unlimited panel of tools and ways to create an orchestra of virtual performers, using DAWs, OSSIA [30], etc. In order to provide a complete usable tool in Max/MSP, we decided to use *bach* [31]. *bach* implements both classical music notation and proportional music notation, with support for accidentals of arbitrary resolution, rhythmic trees and grace notes, polymetric notation, MusicXML, MIDI files, etc. The orchestra simulation with *bach* is implemented in a different patcher than the recognition tool and receives the commands from the automata with OSC. We decided to build fixed contents into the `bach.roll` and `bach.score` objects that allow the user to place notes into a score, just like any score editing program. These objects are then connected to virtual instruments through Midi.

In more general approaches, the recognition program can be connected to a variety of devices that interpret OSC. In our case, the companion program works as follows: each instrument is simulated by a vst that can receive midi notes. The midi notes are sent from the reading of `bach.score` and `bach.roll` objects that each corresponds to a given

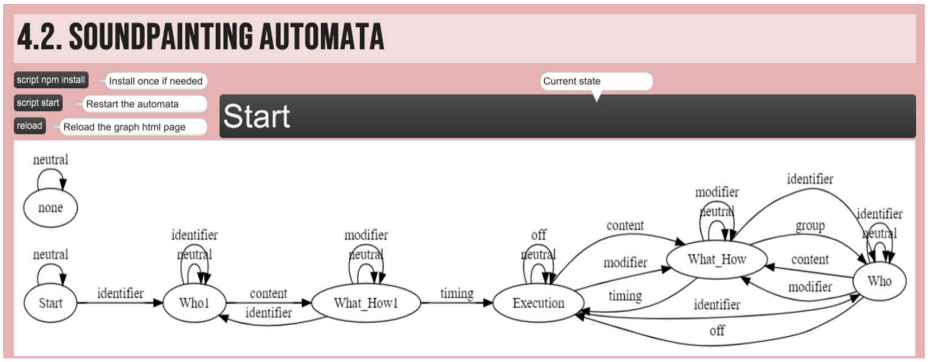


Figure 4. Finite-state automaton representing the grammar for the "default mode" of Soundpainting. The graph shows the different states of the automaton and the possible transitions between them.

musical content of a given instrument and that the user can write and edit just like a normal score. When an OSC command is received from SPQR, a simple regexp routing allow the command to be sent to a given type of musical content (for instance "long tone") of a given instrument (for instance "percussions"). If several long tones are implemented for a single instrument, one is chosen randomly when the command "start" is received. By default, a medium tempo and volume is chosen. But if the commands "tempo" or "volume" followed by a value are received, the `bach.roll` and `bach.score` objects are set to change their values. Therefore, the tempi and volumes of the different instruments can be adjusted in real-time. Once a "stop" command is received, the values of tempo and volume are set back to their default values. This implementation allows us to predefined a set of possible musical elements with Bach objects that are then chosen randomly in the play. Future approaches could rather rely on probabilistic generation models and models of interaction between the different instruments [32].

4. PERFORMANCE AND DEMONSTRATION

The Soundpainting Query Recognition tool is fully available under GPLv3 licence on its Github repository [1]. Detailed informations about the installation process of this SPQR tool can be found on the Github repository as well as in [29].

After installation, even though the number of layers is high, the end-user only manipulates high-level elements on the program: buttons and text elements. Being released by default with webcam inputs, it is easy for the user to run it on a personal computer without any setup on the input part. As for now, we only provide the companion patcher that runs the basic orchestra simulation with *bach* objects. However, the setup of the devices that are controlled at the input and output of SPQR can be modified and added by an experimented users.

The tool was tested using a relatively high-end home computer with 16GB of RAM, an i7-8750H processor and an Nvidia 1060 GTX as a dedicated GPU. For PoseNet, we used the parameters for ResNet50 at quantbytes = 1 and input size = 350, achieving 15 FPS. For Wekinator, we set the default max sequence length to 30, achieving again 15 FPS, which has proven to be more than enough for SPQR. It is to note that PoseNet is very sensitive to light conditions and contrast. Although it is difficult to describe the perfect environment in those terms, the user should pay attention to both camera settings (luminosity or ISO, contrast, saturation profiles, etc) and the position of the lights in the configuration to ensure that the models can work with the best accuracy. A rather uniform background will probably result in good recognition when in high contrast with clothes and skin colour. For our tests, we used two cameras (computer webcam and phone camera) running in parallel. The computer camera is then routed to the full-body model with PoseNet and the phone camera to the model for the hands with HandPose. In general, the choice of the inputs (number, type...) is left to the user and will greatly influence the performance of the recognition. In our case, we worked with a computer camera for simplicity of usage. However, one could reach higher performance and better gesture discrimination with gloves or full body suit 3D tracking systems for instance.

We also released a series of videos, showing the different aspects of this work accessible at `http://deguernel.discordia.fr/spqr/`. The first video explains the gesture and query recognition task using PoseNet, Wekinator DTW, and the SP automaton. We demonstrate the training part of the model using several gestures that are repeated very few times. We compare then the performance of MuBu's gesture follower using Hierarchical Hidden Markov Models and of Wekinator, using DTW, for the classification task, and finally, show how the gesture recognition is used to navigate the SP automaton to form a SP query. The second video shows how to use two cameras (here a webcam and a phone camera) to model both full-body and hand gestures simultaneously with PoseNet and HandPose. Finally, the third video shows a full demonstration of SPQR for music generation using *bach* as a virtual orchestra.

[1] `https://github.com/arthur-parmentier/soundpainting-signs-gestures-recognition`

5. DISCUSSION

From the point of view of the soundpainter, the recognition is an exploratory and creative instrument that can also push for new ways of signing and thinking SP as a direct relation to the instrument (or device) itself. Learning this sign language with a band is not offered to everyone and most orchestras are interested in practising with experimented soundpainters only. In practice, most soundpainters first learn the language as performers before signing themselves, such that they have already internalised most of the language and compositional propositions before endorsing the "role" of soundpainter as a composer.

Feedback is one of the most important aspects of learning. One will for instance learn how to correct himself from errors when he will be able to perceive those as such and understand what the cause of the error is. Even though in the SP design, nothing is considered a mistake on the side of the performer who interprets the sign, the soundpainter can make syntactic mistakes. In a learning tool implementing a grammar, it is important to let the user know why a particular sentence is wrong or what did the program recognize that is not intended. In our SPQR tool, the user can already receive these types of feedback from the automaton, which on top of outputting its actual state and how the request is created also provides error messages that indicate if an unexpected or illegal sign has been observed. For instance, if the user signs "rest of the group, long tone, whole group, minimalism, play", the automaton will output an error message when receiving "whole group, minimalism" , stating that "rest of the group" has already been requested as a content previously in the sentence and that the request of a new content is ambiguous.

Another type of feedback is the ability to hear the contents that are produced by the program and how they react to the different requests, even when the user is making mistakes or the program does not recognize the intended signs. Feedback is one form of incentive to explore more of the tool and learn with it. They can be classified into two categories: external or internal to the program. Some artists are already interested in using the recognition tool in their own installations: they have external incentives. However, some users may not be familiar with digital tools nor with SP and will not take to create something of their own if they are not pushed to it by the program's mechanics that form its internal incentives. Typically, games are good examples of programs with a lot of internal incentives. They have gamified features, such as a score, elements of competition or collaboration, rewards, etc., which push the user to explore more of the game and performing better at it. For future work, there is probably some potential to be explored with SPQR for the implementation of gamified elements or feedback through interactive designs and visualisations. Moreover, we also plan to reach testers to make a more collaborative and musician-oriented development plan and to assess the performances of SPQR more quantitatively. Finally, we also plan to release a compiled version of the system that will simply run as an executable file, simplifying the installation process a lot.

Acknowledgments

A. Parmentier and K. Déguernel share first authorship for this article. This work was made possible thanks to Sarah Kenderdine, the eM+ Lab and the DHLAB at EPFL. We would also like to thank Walter Thompson for his support and the interesting conversations.

6. REFERENCES

[1] W. Thompson, *Soundpainting: the art of live composition. Workbook 1*, 2006.

[2] Guy E. Garnett, "The Aesthetics of Interactive Computer Music," *Computer Music Journal*, vol. 25, no. 1, pp. 21–33, 2001.

[3] D. Keislar, "A historical view of computer music technology," in *The Oxford Handbook of Computer Music*, R. Dean, Ed. Oxford: Oxford University Press, 2011, ch. 2, pp. 11–43.

[4] J.-C. Risset, "Computer music: why," *Songes, Passages, Computer Suite from the Little Boy, Sud. Mainz: Wergo*, vol. 2050, 2013.

[5] D. Herremans, C.-H. Chuan, and E. Chew, "A functional taxonomy of music generation systems," *ACM Computing Surveys*, vol. 50, no. 5, pp. 1–30, 2017.

[6] J. Nika, K. Déguernel, A. Chemla-Romeu-Santos, E. Vincent, and G. Assayag, "DYCI2 agents: merging the "free", "reactive", and "scenario-based" music generation paradigms," in *Proceedings of the International Computer Music Conference*, 2017.

[7] E. A. Edmonds, "Human computer interaction, experience and art," in *Interactive experience in the digital age:evaluating new art practice*, L. Candy and S. Ferguson, Eds. London: Springer, 2014, ch. 2, pp. 11–23.

[8] P. Esling and N. Devis, "Creativity in the era of artificial intelligence," in *Proceedings of the Journées d'Informatique Musicale*, 2020.

[9] F. Negin, P. Rodriguez, M. Koperski, A. Kerboua, J. Gonzàlez, J. Bourgeois, E. Chapoulie, P. Robert, and F. Bremond, "PRAXIS: Towards automatic cognitive assessment using gesture recognition," *Expert Systems with Applications*, vol. 106, pp. 21–35, 2018.

[10] I. Funke, S. Bodenstedt, F. Oehme, F. von Bechtolsheim, J. Weitz, and S. Speidel, "Using 3D convolutional neural networks to learn spatio-temporal features for automatic surgical gesture recognition in video," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2019, pp. 467–475.

[11] X. Gao, Y. Jin, Q. Dou, and P.-A. Heng, "Automatic gesture recognition in robot-assisted surgery with reinforcement learning and tree search," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2020, pp. 8440–8446.

[12] F. Noroozi, D. Kaminska, C. Corneanu, T. Sapinski, S. Escalera, and G. Anbarjafari, "Survey on emotional body gesture recognition," *IEEE Transactions on Affective Computing*, 2018.

[13] M. M. Islam, S. Siddiqua, and J. Afnan, "Real time hand gesture recognition using different algorithms based on american sign language," in *Proceedings of the IEEE International Conference on Imaging, Vision Pattern Recognition*, 2017.

[14] A. R. Jensenius and M. J. Lyons, Eds., *A NIME reader : Fifteen years of New Interfaces for Musical Expression*. Springer, Cham, 2017.

[15] J.-M. Fernández, T. Köppel, G. Lorieux, A. Vert, and P. Spiesser, "GeKiPe, a gesture-based interface for audiovisual performance," in *Proceedings of the New Interfaces for Musical Expression Conference*, 2017, pp. 450–455.

[16] D. Dalmazzo and R. Ramírez, "Air violin: a machine learning approach to fingering gesture recognition," in *Proceedings of the International Workshop on Multimodal Interaction for Education*, 2017, pp. 63–66.

[17] D. Cavdir and G. Wang, "Felt Sound: a shared musical experience for the deaf and hard of hearing," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2020.

[18] D. Chandran and G. Wang, "InterFACE: new faces for musical expression," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2018.

[19] D. V. Nort, I. Jarvis, and M. Palumbo, "Towards a mappable database of emergent gestural meaning," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2016.

[20] D. V. Nort, I. Jarvis, and K. Maraj, "Performing gesture and time via an emergent database," in *Proceedings of the 2020 International Conference on Movement and Computing*, 2020.

[21] T. Pellegrini, P. Guyot, B. Angles, C. Mollaret, and C. Mangou, "Towards Soundpainting gesture recognition," in *Proceedings of the Audio Mostly: A Conference on Interaction With Sound*, 2014.

[22] D. A. G. Jáuregui, I. Dongo, and N. Couture, "Automatic recognition of Soundpainting for the generation of electronic music sounds," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2019.

[23] N. Couture, S. Bottecchia, S. Chaumette, M. Cecconello, J. Rekalde, and M. Desainte-Catherine, "Using the Soundpainting language to fly a swarm of drones," in *Advances in Intelligent Systems and Computing*, J. Chen, Ed. Springer, Cham, 2017, pp. 39–51.

[24] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. P. Murphy, "Towards accurate multi-person pose Estimation in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4903–4911.

[25] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "MediaPipe Hands: On-device real-time hand tracking," in *Proceedings of the Workshop on Computer Vision for Augmented and Virtual Reality*, 2020.

[26] N. Schnell, D. Schwarz, J. Larralde, and R. Borghesi, "PiPo, A plugin interface for afferent data stream processing modules," in *Proceedings of the International Symposium on Music Information Retrieval*, 2017.

[27] R. Fiebrink and P. R. Cook, "The Wekinator: A system for real-time, interactive machine learning in music," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2010.

[28] S. Salvador and P. Chan, "FastDTW: Toward accurate Dynamic Time Warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 70–80, 2004.

[29] A. Parmentier, "Soundpainting Language Recognition," Master's thesis, EPFL, 2020.

[30] J.-M. Celerier, P. Baltazar, C. Bossut, N. Vuaille, J.-M. Couturier, and M. Desainte-Catherine, "OSSIA: Towards a unified interface for scoring time and interaction," in *Proceedings of the International Conference on Technologies for Music Notation and Representation*, 2015.

[31] A. Agostini, D. Ghisi, and J.-L. Giavitto, "Programming in style with bach (extended version)," in *Proceedings of the International Symposium on Computer Music Multidisciplinary Research*, 2019.

[32] K. Déguernel, E. Vincent, and G. Assayag, "Probabilistic factor oracles for multidimensional machine improvisation," *Computer Music Journal*, vol. 42, no. 2, 2018.

LISTEN TO YOUR MIND'S (HE)ART: A SYSTEM FOR AFFECTIVE MUSIC GENERATION VIA BRAIN-COMPUTER INTERFACE

Marco TIRABOSCHI (marco.tiraboschi@unimi.it) (0000-0001-5761-4837) [1],
Federico AVANZINI (federico.avanzini@unimi.it) (0000-0002-1257-5878) [1], and
Giuseppe BOCCIGNONE (giuseppe.boccignone@unimi.it) (0000-0002-5572-0924) [2]

[1] *Laboratorio di Informatica Musicale (LIM)*, *Department of Computer Science*, **Università degli Studi di Milano**, Italy
[2] *PHuSe Lab*, *Department of Computer Science*, **Università degli Studi di Milano**, Italy

ABSTRACT

We present an approach to the problem of real-time generation of music, driven by the affective state of the user, estimated from their electroencephalogram (EEG). This work is aimed at exploring strategies for real-time music generation applications using sensor data. Applications can range from responsive music for x-reality to art installations, and music generation as feedback in pedagogical contexts. We developed a Brain-Computer Interface in the open-source platform OpenViBE. It manages communication with the EEG device and computes the relevant features. A benchmark dataset is used to evaluate the performance of supervised learning methods on the binary classification task of valence and arousal. We also assessed the performance using a reduced number of electrodes and frequency-bands, in order to address the problems of lower budgets and noisy environments. Then, we address the requirements for a real-time music generation model and propose a modification to Magenta's MusicVAE, introducing a parameter for controlling inter-batch memory. In the end, we discuss possible strategies to map desired music features to a model's native input features. We present a Probabilistic Graphical Model to model the mapping from valence/arousal to MusicVAE's latent variables. We also address dataset dimensionality problems proposing three probabilistic solutions.

1. INTRODUCTION

Interest and curiosity towards exploiting the EEG to control sound and music with one's own brain have always been strong in the sound and music community. It was not until 1973 that EEG gained currency as a means for setting a direct communication between brains and computers [1], and it was almost 20 years later that the first BCI successfully allowed the users to control the cursor on a computer screen. But the first reported use of EEG in music is "Music for Solo Performer" by Alvin Lucier (1965) [2]. He had met researcher Edmond Dewan who asked Lucier if he would be interested in using his equipment to detect alpha waves for a piece of music. Alpha waves have frequencies around 8-13 Hz, and would not be audible as audio: so he thought of them as rhythms, thus suitable to create a piece for percussion, by amplifying the alpha bandwidth to drive loudspeakers placed on top of drums membranes [3]. Subsequently, other pioneers of EEG musical applications, such as Richard Teitelbaum, David Rosemboom and Roger Lafosse, exploited brainwaves and other biological signals (e.g. the ECG) to control sound synthesisers, as in the experimental piece "Spacecraft" by Richard Teitelbaum, presented at *Musica Elettronica Viva* in 1967 [4]. In recent years, interest has grown around so-called *Brain-Computer Music Interfaces* [5] and general BCIs because of the increasing affordability of reliable EEG equipment.

The chief concern of this work is to present a system for generating music that reflects the users' affective state, which is estimated from their EEG. We also propose solutions to some practical problems of real BCI systems. Our system is composed of four main modules. Each module is discussed in its own section and sections follow the design order, rather than the data-flow order.

- Brain-Computer Interface (Sec. 3): the EEG hardware and software for acquisition and preprocessing
- EEG Affect Recognition (Sec. 2): the model of valence and arousal correlates of an EEG signal
- Musical Affect Model (Sec. 5): the generative model of musical features conditioned on affective states
- Music Generator (Sec. 4): the model for generating music conditionally to a set of musical features

2. EEG AFFECT RECOGNITION

The EEG Affect Recognition module is the subsystem that is responsible for associating an affective label to the electrical activity of the brain. Affect denotes the mental counterpart of bodily sensation and affective features, such as valence and arousal, capture what a given instance of experience feels like [6]. Valence refers to the feeling of pleasure or displeasure; arousal refers to a feeling of activation or sleepiness. It is worth remarking that in the literature concerning the computational modelling of emotions, the term "affect" is often used interchangeably with that of "emotion" but they should not be confused; emotions are constructed from affect, emotional events being specific instances of affect that are linked to the immediate situation and involve intentions to act [6]. Indeed, the system pre-

sented here deals with affect. However, in what follows, markedly when discussing related work, we will occasionally adopt such convention for the sake of simplicity.

2.1 Related Work

There are several works that address EEG emotion recognition. The DEAP dataset [7] is a popular dataset used as a benchmark for this task. It is a dataset collecting EEG and physiological signals recorded from 32 subjects over 40 trials per subject. The authors also presented some approaches to the emotion recognition task. They used a Gaussian Naive Bayes classifier trained on band-power features. They report a Leave-One-Out Cross-Validation (LOOCV) F_1 score of 0.56 for valence and 0.58 for arousal. Jatupaiboon et al. [8] assessed the problem of real-time valence estimation. They used SVM with power-spectral features for the binary classification of valence. Their work is not directly comparable to the DEAP paper because they use their own dataset, but they show some very important points. First, the average performance of subject-independent models is significantly lower than subject-dependent models (0.65 against 0.75 accuracy). Second, by using only the pair of channels T7 and T8, the performance achieved is comparable to that attained by using all the 14 channels (0.73 accuracy). Menezes et al. exploited the DEAP dataset as a benchmark for emotion recognition in virtual environments [9]. They evaluated band-power features, temporal statistics and the Higher Order Crossing (HOC) features [10] via SVM and Random Forest. They found that band-powers and their statistics performed similarly while the HOC were less predictive.

2.2 Approach

2.2.1 Features

We chose band-powers as features for several reasons: they are the most common features used in EEG emotion recognition; they have a good predictive power; they can be computed efficiently and online; they have been shown to have neurobiological significance [11, 12] in describing the brain activity. For consistency, we adopted the same band definitions as in the DEAP paper: *theta* (4 to 8 Hz), *slow alpha* (8 to 10 Hz), *alpha* (8 to 12 Hz), *beta* (12 to 30 Hz), *gamma* (over 30 Hz). We compute the logarithm of the RMS of the signal for each band and channel. Then, we also compute the difference of each band-power for 14 pairs of symmetric electrodes.

2.2.2 Evaluation Protocol

We divided the 32 subjects randomly into a training set and a test set (60:40). For each subject, every model is evaluated via LOOCV. We will refer to the LOOCV scores on the training subjects as the *validation* scores and to the LOOCV scores on the test subjects as the *test* scores. Comparison between validation scores is performed with related samples tests. The paired-sample T-test [13] for normally distributed samples and the Wilcoxon signed-rank test [14] otherwise. Comparison between validation scores and test scores is performed with an independent samples test, (independent-samples T-test if normal or Wilcoxon rank-sum otherwise). Normality is checked via the Shapiro-Wilk test [15]. We computed validation scores for common machine learning approaches from the BCI literature: Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis [16] (QDA), and Support Vector Machine [17] (SVM). We used QDA with diagonal covariance: this can be called Gaussian Naive Bayes (as in the DEAP dataset paper). As to SVM, we adopted linear, polynomial and RBF kernels. Two "dummy" classifiers were used as reference: one that always predicts the majority class (*Majority*) and another that predicts at random with a probability determined by the label ratio (*Ratio*).

	Valence		Arousal	
Model	mean	std	mean	std
Majority	0.315	0.138	0.310	0.163
SVM (rbf)	0.395	0.133	0.340	0.156
Ratio	0.472	0.062	0.485	0.074
Naive Bayes	0.620	0.095	0.535	0.100
LDA	0.609	0.079	0.560	0.106
SVM (linear)	0.590	0.093	**0.579**	0.078
SVM (poly 7)	**0.626**	0.114	0.557	0.115
Test	0.622	0.092	0.550	0.089

Table 1. Validation F_1 scores for different models and test scores for the best-scoring model (SVM with polynomial kernel of seventh degree) for binary classification of valence and arousal in ascending order of average score. Suboptimal polynomial SVMs are omitted.

2.2.3 Feature Sets

Performance on the full feature set is summarized in Tab. 1. Overall, the results on arousal match the ones in the DEAP dataset paper, but not for valence: we obtained 0.62 accuracy against their 0.56 (this could be due to the slightly different feature set). The best-scoring model (on average and on valence) is the SVM with a 7th degree polynomial kernel. Linear SVM performs better on arousal. Rbf-kernel SVM performs worse than the dummy *Ratio* predictor. The performance of SVM (7th degree polynomial kernel) is significantly different from the dummy predictors and rbf-kernel SVM ($p \leq 0.01$), but not from other models ($p > 0.1$). Test scores are not significantly different from validation.

In a real-time BCI setting, it would be impractical to use an EEG headset with 32 channels, as it would require a very long setup time. The minimal set of channels to be able to use information about band power asymmetry is 2. The brain activity is known to correlate with valence if measured on T7-T8 [8]. Also, activity at CP5-CP6 [7] correlates with arousal. Because of our hardware, we are interested in the specific case of a feature set built using 6 channels. Thus, we evaluated two different setups, adding two pairs of electrodes to either one of the two pairs T7-T8 and CP5-CP6. We wanted one pair at the front of the brain and one at the back, to diversify the information: the pairs FC5-FC6 (frontal-central) and PO3-

6c	T7-T8				CP5-CP6				T7-T8 (α+β)				CP5-CP6 (α+β)			
	Valence		Arousal		Valence		Arousal		Valence		Arousal		Valence		Arousal	
Model	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
LDA	0.512	0.066	0.538	0.113	0.563	0.085	0.522	0.084	0.532	0.074	0.539	0.091	0.550	0.070	0.562	0.104
NB	0.597	0.128	0.525	0.098	**0.622**	0.093	0.531	0.080	0.589	0.118	0.522	0.104	0.603	0.093	0.549	0.086
SVM-L	0.611	0.117	0.553	0.115	0.615	0.101	**0.557**	0.079	0.592	0.123	0.518	0.096	**0.616**	0.083	0.530	0.093
SVM-P	0.606	0.118	0.537	0.111	0.609	0.097	0.550	0.114	0.559	0.106	**0.575**	0.114	0.556	0.092	0.528	0.095

Table 2. Validation F_1 scores for different models on the 6-channels feature sets: using the T7-T8 as central channels or CP5-CP6, and using all frequencies or just *alpha* and *beta* bands. Mean and standard deviation of the F_1 scores are reported for binary classification of valence and arousal.

2c	T7-T8				CP5-CP6				T7-T8 (α+β)				CP5-CP6 (α+β)			
	Valence		Arousal		Valence		Arousal		Valence		Arousal		Valence		Arousal	
Model	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
LDA	0.601	0.125	0.503	0.128	**0.612**	0.060	**0.555**	0.092	0.587	0.113	0.487	0.092	0.617	0.087	0.514	0.099
NB	0.584	0.131	0.509	0.105	0.610	0.071	0.529	0.094	0.561	0.137	0.487	0.078	**0.593**	0.092	0.544	0.087
SVM-L	0.619	0.136	0.494	0.150	0.640	0.072	0.535	0.104	0.578	0.171	0.435	0.073	0.571	0.122	0.485	0.123
SVM-P	0.650	0.070	0.515	0.086	0.602	0.101	0.538	0.090	0.577	0.119	0.529	0.119	0.561	0.114	0.512	0.105

Table 3. Validation F_1 scores for different models on the 2-channels feature sets: using the T7-T8 channels or CP5-CP6, and using all frequencies or just *alpha* and *beta* bands. Mean and standard deviation of the F_1 scores are reported for binary classification of valence and arousal.

PO4 (parieto-occipital) show correlations with affective labels [7]. We evaluated performance on two 6-channels feature sets: FC5, FC6, PO3, PO4 with T7-T8 or with CP5-CP6. In real-world operation, low-frequency components can be subject to noise from muscle movement. Also, high-frequency components can be affected by power-line interference (50 Hz or 60 Hz). Hence, we also assessed the performance of the models trained only on the central frequency bands (*slow alpha*, *alpha* and *beta*). Validation scores on 6-channel feature sets are summarized in Tab. 2 for all four configurations: two channel choices, both with all frequency bands or only with central frequency bands (α+β). We also assessed the performance on 2-channels feature sets, using T7-T8 or CP5-CP6, exploiting all frequency bands or just the central bands (Tab. 3).

2.3 Results

Using 6 channels, F_1-scores for valence classification are very similar to the ones obtained with the full feature set. Scores on arousal slightly decreased. Using all frequency bands, the best model on average is the Linear SVM (0.62 on valence and 0.56 on arousal). Using only the central bands, the best model on average is Naive Bayes (0.60 on valence and 0.55 on arousal). In almost every case, performance is better using CP5-CP6. Employing 2 channels with all frequency bands, F_1-scores are still very similar to the previous ones. However, the two best validation scores on valence (Polynomial SVM with T7-T8 and Linear SVM with CP5-CP6) are significantly different from the test scores: the test score for Linear SVM is 0.54 ($p < 0.05$). Thus, we consider LDA as the best model (0.61 on valence and 0.56 on arousal), since it is consistent across the two partitions. We observed that performance on arousal using T7-T8 is not significantly different from the dummy predictor (Ratio). We surmise that T7-T8 is not a sufficient configuration for arousal classification, in contrast to its use for valence classification (as in previous literature [8]). Using 2 channels and only the central bands, none of the models is significantly better than the dummy predictors for arousal classification (all $p < 0.05$). Valence classification is still possible, but the validation score of LDA with CP5-CP6 (0.62) is significantly different from the test score: 0.52 ($p < 0.01$). So, we consider Naive Bayes as the best model, with an F_1-score of 0.59.

Based on these observations, we determined three different setups to address different requirements: FC5-FC6-CP5-CP6-PO3-PO4 (α+β) for robust features, CP5-CP6 for minimal hardware and CP5-CP6 (α+β) for robust features and minimal hardware, but for valence only.

3. BRAIN-COMPUTER INTERFACE

The Brain-Computer Interface module is the system that allows for sensing the brain activity of the user and extracting the relevant features.

3.1 Related Work

Brain-Computer Interfaces for music are becoming more and more popular. BCIs are often categorized as "passive" (using arbitrary brain activity without the purpose of voluntary control), "reactive" (using brain activity arising in reaction to external stimulation), and "active" (using brain activity that is consciously controlled by the user, independently from external events) [18]. A popular technique for reactive BCIs is based on *steady state visually evoked potentials* (SSVEP). It consists in presenting images on screen that flicker at different rates, and detecting electrical potentials on the visual cortex to determine which object the user is watching. SSVEP-BCIs have been used for

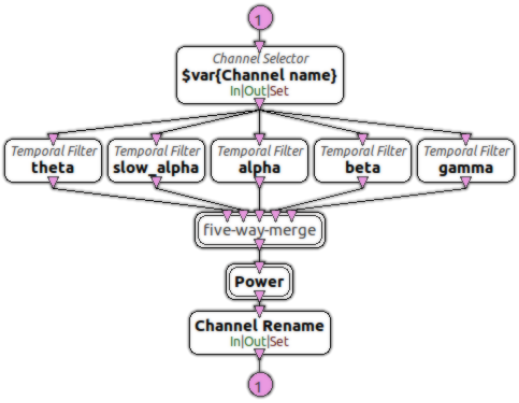


Figure 1. The *Band-powers DEAP* metabox. This metabox selects one channel (named *Channel name*) out of the multi-channel signal and computes its band-powers.

music writing, music navigation [19], and sonic expression [20]. A common approach to active BCIs is *motor imagery*. It consists in detecting patterns correlated to the imagination of motor activity [21] (μ *rhythms*). MI-BCIs are becoming popular for video game control [22]. Passive BCIs are often exploited in affective computing, e.g. for monitoring attention, stress and affective states [23].

3.2 Approach

We used the g.tec g.Sahara active dry EEG electrode system headset. We developed a feature extraction software for the BCI in OpenViBE, a cross-platform open-source environment [24]. It is headset-independent, because the OpenViBE *acquisition server* handles incoming signals. Thus, it can be used with any EEG headset, provided that the drivers are available. The OpenViBE designer is a visual programming environment. An executable is called a *scenario* and its components *boxes*. A scenario can be used within another scenario as a *metabox*.

3.2.1 Feature Extraction

For each channel, we compute the band-powers. We defined the band-power of a frequency band as the mean-square of the band-passed signal (see Sec. 2.2.1). We defined the metabox *Power* to compute the power of a signal over overlapping temporal windows. Then, we developed the metabox *Band-powers DEAP*, that selects one channel and splits it into 5 different bands using a time-domain filterbank. It uses the *Power* metabox to compute the band-powers. The band-power signals are rearranged into a single multi-channel signal and renamed. The metabox *Band-Power Features* instantiates a *Band-powers DEAP* metabox for each channel to extract its band-powers and rearranges the band-power signals into a single multi-channel signal.

3.2.2 Data Transmission

After computing the band-power features, we need to be send them to the EEG affect recognition process. We use the Open Sound Control (OSC) [25] network protocol for this. Due to the great interest of the sound and music community in BCIs, OSC has also become a common protocol for BCIs and some companies that develop BCIs provide OSC utilities, such as Emotiv's *Mind Your OSCs*. OpenViBE also has a rudimentary OSC client *box*. Each feature is sent on a different OSC method. The address pattern is `/eeg/<channel>/<band>`.

4. ONLINE MUSIC GENERATION

The Online Music Generation module is the system that generates music in real-time. We want to be able to control it with parameters that can change over time. We propose a transfer-learning approach for generating music via affective parameters.

4.1 Related Work

The recent developments in deep learning enabled new approaches that are now the state-of-the art of music generation. Deep learning for music mainly exploits advancements in natural language modelling. Especially, the introduction of *attention-based* Recurrent Neural Networks [26] (RNN) allowed for longer time-scale coherence than before. The Magenta project by Google Brain has developed several deep learning models for computational creativity. Melody RNN [27] employs an attention-based LSTM (*Long Short-Term Memory*, a type of RNN) for generating melodies. To improve long-term structure, they developed a hierarchical RNN decoder for an autoencoder called MusicVAE [28]. The latest architecture in language modelling is the transformer [29], a deep neural network that does not use recurrence, but relies entirely on attention, such as OpenAI's GPT-2 [30] and GPT-3 [31]. Magenta's Music Transformer [32] is a transformer that employs relative self-attention for music generation. OpenAI's MuseNet [33] is based on their GPT-2 and is a large-scale transformer for symbolic generation that supports up to 10 different instruments. On the other hand, their Jukebox [34] generates raw musical audio of fully arranged compositions with singing voice. It can be conditioned on either artist, genre or lyrics. The main drawback of deep learning is the amount of data required for training: as an example, Music Transformer has been trained with 10 000 hours of piano music retrieved from YouTube and converted from audio to MIDI by another neural network, Onsets and Frames [35].

4.2 Modified MusicVAE

We chose MusicVAE for real-time music generation. One reason is that pre-trained checkpoints are available for download. Also, operations in its latent space have semantic effects on the output (e.g. interpolation in the latent space results in the semantic interpolation of the MIDI content). Finally, it supports multi-instrument pieces. A pre-trained checkpoint is available for use with *trios* (drums, bass and melody). We propose a small modification that does not require re-training for real-time parameter modification. MusicVAE is a variational auto-encoder (VAE)

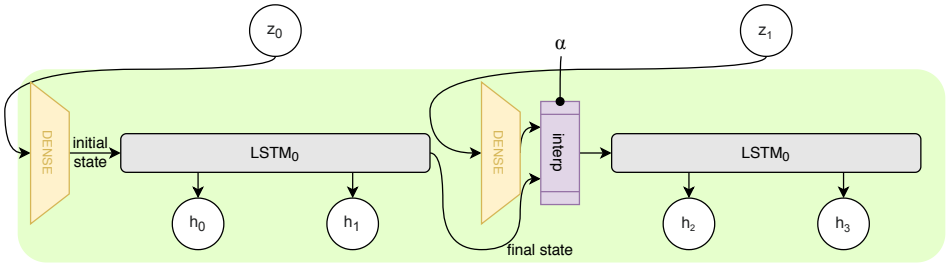


Figure 2. Top layer of the restructured decoder RNN for online music generation. The final state of the LSTM is interpolated with the new state to obtain a new initial state that is *in-between* the two. This results in a transition between the two parts.

composed of a deep Bidirectional LSTM (BLSTM) probabilistic encoder and a hierarchical LSTM probabilistic decoder. The encoder encodes an entire input MIDI into one single latent vector. It can only be executed batch because of its bidirectionality (the output at any time step depends both from past and future inputs). The decoder decodes a latent vector into any length of musical content. It can be executed in real-time because it is monodirectional. Also, memory cost does not increase with time because the output depends only on the current state of the network. The drawback of MusicVAE for our intended application is that the decoder doesn't allow any input for transitioning to a different part without starting over. Thus, we restructured the network for sequentially generating music with smooth transitions. The first layer of the network is visualized in Fig. 2. When sampling conditionally on a new encoding z_{t+1}, we compute the corresponding new initial state for the RNN using the same fully-connected layer as the original network. We then compute the new state for the LSTM as a convex combination of that output and the final state from the previous execution of the LSTM. We introduce a *memory* coefficient $\alpha \in [0, 1]$. Defining f_t the final state after decoding z_t and $d(z_t)$ the result of applying the dense layer to z_t, the new initial state will be

$$s_{t+1} := \alpha \cdot f_t + (1 - \alpha) \cdot d(z_{t+1}). \quad (1)$$

Setting $\alpha = 0$ results in independent samples (the network forgets the previous state), whilst setting $\alpha = 1$ the network ignores new inputs. Setting the memory to an in-between value allows for adjusting the trade-off between coherence ($\alpha \to 1$) and change ($\alpha \to 0$). We implemented this modification in Python by extending Magenta's own class for MusicVAE pre-trained models and overriding the definition of the decoding operation.

4.3 Multiprocess System

Decoding a MIDI section from MusicVAE is not a fast operation in a musical context. It can take several seconds on a laptop using a CPU. We developed a dual-process system to overcome this problem. The two processes involved are a MIDI sequencer client and a MusicVAE server. Since they are local processes, we handle their communication with pipes. We implemented the MIDI sequencer using the Python bindings for FluidSynth [36]. The sequencer is never blocked waiting for a request of a MIDI sequence. Instead, every time the client callback is called, if there is a pending request, the MIDI is read from the pipe and scheduled for the synthesizer. Then, a new MIDI sequence is requested from the client.

5. MUSICAL AFFECT MODEL

The module named Musical Affect Model is the subsystem that maps musical features to affective labels. Here, musical features are encoding vectors in the MusicVAE latent space (see Sec. 4) and an affective state is a pair of binary labels for valence and arousal (see Sec. 2).

5.1 Related Work

Affect correlates of music features have always been a subject of great interest for musicologists, although they are not as often computationally exploited for music generation. Williams *et al.* propose a taxonomy for what they refer to as Affective Algorithmic Composition (AAC) systems [37]. AAC systems can be compositional if they generate music (e.g. Robertson *et al.* [38]) or performative if they execute a musical piece in a way that reflects the target emotion (such as RaPScoM [39]). Briefly, they are generative if they write new music or transformative if they modify a given input, such as a music production system [40]. They can be real-time or in batch: real-time systems are adaptive if they can adjust their output during execution, as our modified MusicVAE (see Sec. 4.2). Williams *et al.* later presented an AAC system that targeted affective states by means of lookup table of musical features compiled from literature review [41]. They specify a discrete mapping from the affective-state space to the set of musical features, then a neural network outputs MIDI. Kirke and Miranda developed an AAC system for communicating the affective state detected from an EEG signal [42]. Valence is computed as the difference of logarithmic alpha-band energy between left and right frontal regions of the brain. Arousal is computed as the negative of the sum of such logarithmic alpha-band energies. Binary valence and arousal are used to transform a pre-written musical score by changing key, pitch and tempo. Galvanic skin response (GSR) is another biological signal that is known to correlate with affective states and Daly *et al.* developed a system where GSR serves as input for affective music generation [43].

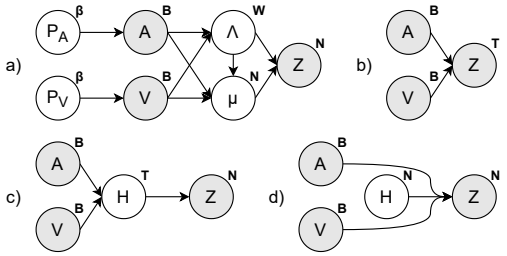


Figure 3. Probabilistic Graphical Models discussed in text. A PGM is a directed graph where nodes represent random variables and arcs stand for conditional probabilities. Hidden variables are in white and observed variables are in grey. Annotations on top of each variable denote the type of its conditional distribution (Beta, Bernoulli, Wishart, Normal or T). The PGM (a) is simplified in a graph only including observed variables (b). PPCA introduces a new hidden variable H: PPCA can be performed on the entire dataset (c) or for each class (d). See text for details.

5.2 Approach

We developed a directed Probabilistic Graphical Model (PGM) to map affective labels to MusicVAE encodings.

5.2.1 Dataset

The *MIREX-like mood* dataset is a dataset for multimodal music emotion recognition [44]. It collects 903 audio samples, 193 of which with lyrics and MIDI. Affective tags are adjectives, grouped in 5 clusters. First, we preprocessed the dataset to get the MusicVAE encodings from the MIDI files: 151 of the 193 MIDI files were compatible with the MusicVAE *trio* model. We used a dataset containing the "*Norms of valence, arousal, and dominance for 13 915 English lemmas*" [45] for converting adjectives to valence and arousal values. We observed that the 5 pre-defined clusters did not map to clusters in the valence-arousal 2-D space. We partitioned the samples into four classes by binarizing valence and arousal, median values being the thresholds.

5.2.2 Model

We used a directed PGM to model the interdependency of the different variables. As a consequence of MusicVAE's ELBO loss function, the prior distribution of the latent codes Z is a standard *multivariate Normal distribution* (MVN) [28]. Therefore, we model the conditional distribution of Z given an affective state (a, v) as a MVN, as well. The mean and precision parameters given each affective state $(\mu_{a,v}, \Lambda_{a,v})$ are unknown. The joint posterior distribution of the unknown mean and precision parameters of a MVN is a Normal-Wishart distribution. However, they are never observed and we are not interested in inferring them. So, we can directly model the distribution of latent codes given an affective state as a multivariate Student's T distribution: this is the distribution of the samples of a MVN whose mean and the precision are Normal-Wishart distributed. Valence and arousal only assume binary values, so, we model them as Bernoulli variables. The mean parameters (p_a and p_v) are unknown. The posterior distribution of the mean of a Bernoulli variable is a Beta distribution. As previously, their inference is not of interest. The distribution of a Bernoulli variable whose mean is Beta distributed is still a Bernoulli distribution. The full PGM and its simplified version are visualized in Fig. 3a and Fig. 3b.

5.2.3 Dimensionality Reduction

The dimensionality of the latent space is much larger than the available data points. The MusicVAE trio model has 512 latent variables and only 151 points are in the dataset: when partitioned into 4 classes, it amounts to an average of 38 points per class. This is not sufficient for inferring the parameters of the multivariate Student's T distribution because the sample covariance matrix is singular. We present three approaches for overcoming this problem.

We can make a Naive Bayes assumption, imposing all features to be independent from each other given the class. Often Naive Bayes is applied in contexts where the independence assumption is not supported [46]. In our case, it could be partially motivated by the fact that the prior distribution of the encodings is a standard MVN, for which the assumption holds. We applied Naive Bayes to our graphical model by setting to zero all non-diagonal values of the sample covariance matrix.

We can use probabilistic PCA (PPCA) [47] to map our samples to a lower-dimensional space. There are two possible ways of applying PPCA to our graph. We can use a *class-independent* PPCA to map all the latent codes to a lower-dimensional space, where we can estimate the parameters of the class-wise T distributions. The resulting PGM is visualized in Fig. 3c. Alternatively, we can use PPCA as the model of the distribution of the latent codes of each class. In this setting, each affective state corresponds to different values for the PPCA parameters. The corresponding BBN is shown in Fig. 3d. [1]

6. CONCLUSIONS

We have presented a pipeline for generating affectively-driven music using the EEG. Main results so far achieved can be recapped as follows. We have shown that a reduced number of EEG channels can still be effective for the binary classification of valence and arousal, resulting in cheaper and more practical BCIs. We have also developed an online feature extraction algorithm using the OpenViBE platform. This software is cross-platform and headset independent. We used the OSC protocol for the communication with the affect classification module. We discussed the requirements for an online music generation algorithm and made a modification to a pre-trained neural autoencoder (MusicVAE). This modification allows control over the latent codes when generating music, so that the output MIDI is the result of a trajectory in the latent space, instead of a single static code. Finally, we proposed a probabilistic model for mapping affective labels to music features, in the

[1] Examples generated with a *class-dependent* PPCA (6 principal components) are publicly available at `https://chromaticisobar.github.io/ListenToYourMindsHeArt`

form of MusicVAE latent codes. Thee different probabilistic solutions are presented for dimensionality problems.

In the future, we plan to make a thorough investigation in the relationship between the number of EEG channels and the affect classification performance. Also, we plan to collect a dataset of affectively labelled MIDI to evaluate the dimensionality reduction models we exploited when mapping affective labels to music features.

Acknowledgments

Part of this work has been possible thanks to the collaboration with the *Interdisciplinary Centre for Computer Music Research* of University of Plymouth. This collaboration was funded by "Thesis Abroad" scholarship of University of Milan. We would especially like to thank Prof. Eduardo Miranda, Dr. Edward Braund and Mr. Satvik Venkatesh.

7. REFERENCES

[1] J. J. Vidal, "Toward direct brain-computer communication," *Annual Review of Biophysics and Bioengineering*, 1973.

[2] V. Straebel and W. Thoben, "Alvin Lucier's music for solo performer: experimental music beyond sonification," *Organised sound*, 2014.

[3] V. Rusche and H. Harder, "No ideas but in things: The composer Alvin Lucier," Movie, 2012.

[4] B. Arslan, A. Brouse, J. Castet, J.-J. Filatriau, R. Lehembre, Q. Noirhomme, and C. Simon, "From biological signals to music," in *2nd International conference on enactive interfaces*, 2005.

[5] E. R. Miranda and J. Castet, *Guide to brain-computer music interfacing*. Springer, 2014.

[6] L. F. Barrett, "The theory of constructed emotion: an active inference account of interoception and categorization," *Social cognitive and affective neuroscience*, vol. 12, no. 1, pp. 1–23, 2017.

[7] S. Koelstra, C. Muhl, M. Soleymani, J.-S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, and I. Patras, "Deap: A database for emotion analysis; using physiological signals," *IEEE transactions on affective computing*, vol. 3, no. 1, pp. 18–31, 2011.

[8] N. Jatupaiboon, S. Pan-ngum, and P. Israsena, "Real-time eeg-based happiness detection system," *The Scientific World Journal*, vol. 2013, 2013.

[9] M. L. R. Menezes, A. Samara, L. Galway, A. Sant'Anna, A. Verikas, F. Alonso-Fernandez, H. Wang, and R. Bond, "Towards emotion recognition for virtual environments: an evaluation of eeg features on benchmark dataset," *Personal and Ubiquitous Computing*, vol. 21, no. 6, pp. 1003–1013, 2017.

[10] P. C. Petrantonakis and L. J. Hadjileontiadis, "Emotion recognition from eeg using higher order crossings," *IEEE Transactions on information Technology in Biomedicine*, vol. 14, no. 2, pp. 186–197, 2009.

[11] G. Deuschl and A. Eisen, *Recommendations for the practice of clinical neurophysiology: guidelines of the International Federation of Clinical Neurophysiology*. Elsevier Health Sciences, 1999, no. 52.

[12] R. Yuvaraj, M. Murugappan, N. M. Ibrahim, M. I. Omar, K. Sundaraj, K. Mohamad, R. Palaniappan, E. Mesquita, and M. Satiyan, "On the analysis of eeg power, frequency and asymmetry in parkinson's disease during emotion processing," *Behavioral and brain functions*, vol. 10, no. 1, p. 12, 2014.

[13] Student, "The probable error of a mean," *Biometrika*, pp. 1–25, 1908.

[14] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.

[15] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.

[16] D. Garrett, D. A. Peterson, C. W. Anderson, and M. H. Thaut, "Comparison of linear, nonlinear, and feature selection methods for eeg signal classification," *IEEE Transactions on neural systems and rehabilitation engineering*, vol. 11, no. 2, pp. 141–144, 2003.

[17] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, pp. 1–27, 2011.

[18] T. O. Zander, C. Kothe, S. Jatzev, and M. Gaertner, "Enhancing human-computer interaction with input from active and passive brain-computer interfaces," in *Brain-Computer Interfaces - Applying Our Minds to Human-Computer Interaction*, D. S. Tan and A. Nijholt, Eds. Springer, 2010, pp. 181–199.

[19] S. Venkatesh, "Investigation into stand-alone brain-computer interfaces for musical applications," Master's thesis, University of Plymouth, 2019.

[20] S. Venkatesh, E. Braund, and E. R. Miranda, "Designing brain-computer interfaces for sonic expression," in *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Birmingham City University, 2020, pp. 525–530.

[21] A. S. Aghaei, M. S. Mahanta, and K. N. Plataniotis, "Separable common spatio-spectral patterns for motor imagery bci systems," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 1, pp. 15–29, 2015.

[22] D. Coyle, J. Garcia, A. R. Satti, and T. M. McGinnity, "Eeg-based continuous control of a game using a 3 channel motor imagery bci: Bci game," in *2011 IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB)*. IEEE, 2011, pp. 1–7.

[23] A.-M. Brouwer, J. van Erp, D. Heylen, O. Jensen, and M. Poel, "Effortless passive bcis for healthy users," in *Universal Access in Human-Computer Interaction. Design Methods, Tools, and Interaction Techniques for eInclusion*, C. Stephanidis and M. Antona, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 615–622.

[24] Y. Renard, F. Lotte, G. Gibert, M. Congedo, E. Maby, V. Delannoy, O. Bertrand, and A. Lécuyer, "Openvibe: An open-source software platform to design, test, and use brain–computer interfaces in real and virtual environments," *Presence: teleoperators and virtual environments*, vol. 19, no. 1, pp. 35–53, 2010.

[25] A. Freed, "Open sound control: A new protocol for communicating with sound synthesizers," in *International Computer Music Conference (ICMC)*, 1997.

[26] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[27] E. Waite *et al.*, "Generating long-term structure in songs and stories," *Web blog post. Magenta*, vol. 15, 2016.

[28] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A hierarchical latent vector model for learning long-term structure in music," *arXiv preprint arXiv:1803.05428*, 2018.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[30] A. Radford, J. Wu, D. Amodei, D. Amodei, J. Clark, M. Brundage, and I. Sutskever, "Better language models and their implications," *OpenAI Blog https://openai. com/blog/better-language-models*, 2019.

[31] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[32] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer," *arXiv preprint arXiv:1809.04281*, 2018.

[33] C. M. Payne, "Musenet," OpenAI Blog, 2019.

[34] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, "Jukebox: A generative model for music," *arXiv preprint arXiv:2005.00341*, 2020.

[35] C. Hawthorne, E. Elsen, J. Song, A. Roberts, I. Simon, C. Raffel, J. Engel, S. Oore, and D. Eck, "Onsets and frames: Dual-objective piano transcription," *arXiv preprint arXiv:1710.11153*, 2017.

[36] J. Newmarch, "Fluidsynth," in *Linux Sound Programming*. Springer, 2017, pp. 351–353.

[37] D. Williams, A. Kirke, E. R. Miranda, E. B. Roesch, and S. J. Nasuto, "Towards affective algorithmic composition," in *The 3rd International Conference on Music & Emotion, Jyväskylä, Finland, June 11-15, 2013*. University of Jyväskylä, Department of Music, 2013.

[38] J. Robertson, A. de Quincey, T. Stapleford, and G. Wiggins, "Real-time music generation for a virtual environment," in *Proceedings of ECAI-98 Workshop on AI/Alife and Entertainment*. Citeseer, 1998.

[39] J. Doppler, J. Rubisch, M. Jaksche, and H. Raffaseder, "Rapscom: towards composition strategies in a rapid score music prototyping framework," in *Proceedings of the 6th Audio Mostly Conference: A Conference on Interaction with Sound*, 2011, pp. 8–14.

[40] A. P. Oliveira and A. Cardoso, "Automatic manipulation of music to express desired emotions," in *Proceedings of the 6th Sound and Music Computing Conference. Porto: University of Porto*. Citeseer, 2009, pp. 265–270.

[41] D. Williams, A. Kirke, E. Miranda, I. Daly, F. Hwang, J. Weaver, and S. Nasuto, "Affective calibration of musical feature sets in an emotionally intelligent music composition system," *ACM Transactions on Applied Perception (TAP)*, vol. 14, no. 3, pp. 1–13, 2017.

[42] A. Kirke and E. R. Miranda, "Combining eeg frontal asymmetry studies with affective algorithmic composition and expressive performance models," in *ICMC*, 2011.

[43] I. Daly, A. Malik, J. Weaver, F. Hwang, S. J. Nasuto, D. Williams, A. Kirke, and E. Miranda, "Towards human-computer music interaction: Evaluation of an affectively-driven music generator via galvanic skin response measures," in *2015 7th Computer Science and Electronic Engineering Conference (CEEC)*. IEEE, 2015, pp. 87–92.

[44] R. Panda, R. Malheiro, B. Rocha, A. Oliveira, and R. P. Paiva, "Multi-modal music emotion recognition: A new dataset, methodology and comparative analysis," in *International Symposium on Computer Music Multidisciplinary Research*, 2013.

[45] A. B. Warriner, V. Kuperman, and M. Brysbaert, "Norms of valence, arousal, and dominance for 13,915 english lemmas," *Behavior research methods*, vol. 45, no. 4, pp. 1191–1207, 2013.

[46] H. Zhang, "The optimality of naive bayes," *AA*, vol. 1, no. 2, p. 3, 2004.

[47] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

FB1_ODE – AN INTERFACE FOR SYNTHESIS AND SOUND PROCESSING WITH ORDINARY DIFFERENTIAL EQUATIONS IN SUPERCOLLIDER

Daniel MAYER (mayer@iem.at)[1]

[1] *Institute of Electronic Music and Acoustics (IEM)*, **University of Music and Performing Arts,** Graz, Austria

ABSTRACT

The class Fb1_ODE, included in the miSCellaneous_lib quark extension library [1] of SuperCollider (SC, [2, 3]), enables the audible integration of ordinary (systems of) differential equations (ODEs) with initial values in realtime. The prefix 'Fb1' refers to the class Fb1 for single sample feedback and feedforward, on which it depends [4]. Consequently, the numerical integration of ODE systems with a step width of one sample is possible with arbitrary block sizes of SC's audio engine. Fb1_ODE opens the possibility for immediate audio experiments with models from physics, electrical engineering, population dynamics, chemistry, etc., preferably those with oscillatory respectively quasi-oscillatory solutions or chaotic features. Designing new ODEs from scratch or altering respectively disturbing systems can also be interesting regarding the sounding results. Wrappers of Fb1_ODE include well-known systems like Van der Pol, Duffing, Hopf, Mass-Spring-Damper, and Lorenz; users can interactively add other systems with the class Fb1_ODEdef. The modulation of ODE parameters, system time, and the feeding of additional audio signals into ODE systems are, amongst others, further options for unorthodox synthesis with differential equations.

1. INTRODUCTION

We regard systems of the form

$$Y'(t) = F(t,\ Y(t)) \tag{1}$$

in the domain of real numbers where Y and F can be vector-valued functions and the restriction of an initial value condition

$$Y(t_0) = y_0 \tag{2}$$

In a physical interpretation, t is the system time.

1.1 Why using ODEs for audio synthesis?

That is a legitimate question, not at least because of some counterarguments. There are numerical hurdles, calculations often become CPU-demanding, and the usage of arbitrary ODE models in many fields of science and technology is no direct argument for their application in sound and music – besides from the ongoing research in acoustic and physical models [5]. However, an outweighing argument for comprehensive ODEs also comes from the fact that they can work as a generic **description system** for waveforms: many ODE solutions cannot be expressed in an analytical form. This consideration leads to the assumption that ODEs can act as a key to a land of unknown and intriguing possibilities in sound synthesis. The growing significance – one might even say: popularity – of non-linear dynamical systems has certainly supported this view. How to choose from these possibilities in artistic regard is a crucial question that needs practical exploration – a general-purpose tool as the presented one aims to provide quick feedback.

The power of ODEs as a description system already shows up by this trivial example. The second-order differential equation

$$y''(t) = -y(t) \tag{3}$$

is – by the substitution

$$w(t) = y'(t) \tag{4}$$

– reduced to the first-order ODE system

$$\begin{aligned} y'(t) &= w(t) \\ w'(t) &= -y(t) \end{aligned} \tag{5}$$

With the initial values $y(0) = 0$ and $w(0) = 1$, the system has the solution

$$\begin{aligned} y(t) &= \sin(t) \\ w(t) &= \cos(t) \end{aligned} \tag{6}$$

The initial equation (3) is, obviously, much more compact than the representations of sine and cosine as infinite series derived from Taylor expansion. On the other hand, using numerical ODE integration for producing a sine

wave doesn't make much sense except a proof of concept. However, there exist many ODE systems with brief definitions that can generate rich and evolving spectra.

Researchers have made several suggestions to use ODEs for audio synthesis and processing. The approaches partially origin from the world of analog circuits (Slater [6]: Ueda, a variant of the Duffing oscillator, Choi [7] and Rodet [8]: Chua circuit). Jacobs [9] uses the FitzHugh-Naguamo model of which Van Der Pol is a particular case. See Falaize and Hélie for stable simulations of analog audio circuits from electronic schematics [10]. State space models aim to represent analog systems by input, output and state variables as parts of ODE systems (e.g., [11, 12]). Wave digital filters are an attempt to digitize analog circuits by ODEs and traveling-wave components [13].

SC's main class library already includes the famous and widely used Lorenz model. The implementation via Fb1_ODE additionally allows the feeding of external input into the system, an experimental feature of ODE synthesis also recommended by Stefanakis, Abel, and Bergner [14, pp. 53–55, 57].

1.2 Numerical integration of ODEs

There exist many integration techniques, which serve well in typical engineering applications. However, there is a specific demand with the audification of ODEs: oscillations should be kept stable over a relatively long period – or put in other words, we need many oscillations to get an audible signal for a significant amount of time. It is likely to encounter drifts in the long run with arbitrary integration methods. E.g., already in the case of a simple harmonic oscillator, a 3rd order Runge-Kutta scheme can fail (Figure 5). For several oscillators, there exist specific integration schemes (Duffing: Bilbao [5, p. 75–77], Van der Pol: [15]). More general, so-called symplectic procedures have gained attention [16]. Roughly spoken, they preserve volume in a geometric sense and are often well suited for ODE audification. See the chapter on integration in David Pirrò's dissertation [17, pp. 135–146]. David Pirrò has implemented the symmetric symplectic "rattle" integration in his optimized ODE compiler *Henri*, Fb1_ODE uses this scheme as default [18, 19, 20]. Implicit integration techniques are also used widely in audio applications [21]. For implementational reasons – especially the interactive adding of schemes – preference has been given to the explicit methods, from which several are built-in at choice (see 2.5).

2. IMPLEMENTATION IN SC

2.1 Definition of ODEs and usage as unit generators

The user interface for the most general purposes mainly consists of two classes in the SC language (the SC client): Fb1_ODEdef and Fb1_ODE. After defining an ODE with Fb1_ODEdef – by providing the function F of (1) as an SC Function – it is ready for synthesis usage with Fb1_ODE. The latter is a so-called pseudo-UGen (pseudo unit generator), a compound UGen structure comparable to macros in other languages. Under the hood, Fb1_ODE merges the selected numerical procedure, applied to F, into a UGen graph. That ensures integration on a per-sample base when employing the compiled synthdef (instrument) on the SC server (the audio engine). It also holds for a server block size greater than 1 by involving the single sample feedback pseudo-UGen Fb1 [4]. Figure 1, as a proof of concept, shows the code for producing a sine wave by using the harmonic oscillator system.

```
// The Function (curly brackets) must take time and
// an array as args and return an array of the same size (2).
// In contrast to math notation indices start from 0.

Fb1_ODEdef(\harmonic,
    { arg t, y; [y[1], y[0].neg] },
    // default init values t0, y0
    t0: 0, y0: [0, 1]
)

// The stereo output is a pair of 90-degree-shifted sines.
// As one wavelength would be 2pi seconds, multiply by
// a factor 500 * 2pi to get 500 Hz.
// blockSize is detected automatically.

x = { Fb1_ODE.ar(\harmonic, tMul: 500 * 2pi) * 0.1 }.play
```

Figure 1. Sine wave by ODE integration

More interesting, basic systems like the harmonic oscillator or exponential decay can be the starting point for experimental variations. It's a promising strategy to gradually drift away from a base case, e.g., define the system

$$\begin{aligned} y'(t) &= w(t) \\ w'(t) &= -y(t)\,(1 + k\,w(t)) \end{aligned} \tag{7}$$

With $k = 0$, it equals the harmonic oscillator, greater values produce a brass-like sound.

```
// define system function with k as additional arg
Fb1_ODEdef(\harmonic_ext_1, { arg t, y, k;
    [
       y[1],
       y[0].neg * (1 + (k * y[1]))
    ]
}, 0, [0, 1]);

x = {
    Fb1_ODE.ar(
        \harmonic_ext_1,
        [2], 1500, // k = 2, tMul = 1500
        0, [0, 1] // init values
    ) * 0.1;
}.play;
```

Figure 2. Blurred harmonic oscillator.

2.2 Wrapper classes

The library provides dedicated pseudo-UGen classes for the well-known systems Van der Pol, Duffing, Hopf, Lorenz, and Mass-Spring-Damper (MSD). The latter satisfies the second-order differential equation

$$m\,y''(t) = -k\,y(t) - c\,y'(t) + F(t) \tag{8}$$

whereby m denotes the mass, c the dampen factor, k the spring stiffness, and $F(t)$ the externally applied force. Like

the harmonic oscillator, one can transform it into a system of two first-order equations whose solutions describe position and velocity. The code example in Figure 3 uses the oscillation, which converges to an end position, for frequency modulation. The audible result is a timbral development to a sine with a fixed frequency. Note that, by default, Fb1_ODE and the wrapper classes apply a DC leaker, which, in this case, is disabled.

```
x = {
    var f = 0.2, mass = 0.001, spring = 50,
        dampen = 0.0005, sig, mod;
    sig = Fb1_MSD.ar(
        f, mass, spring, dampen,
        // deflection converges to 0.004
        // hence take no LeakDC
        leakDC: false
    );
    // map position to useful frequencies
    // LR-decorrelation by slightly different ranges
    mod = sig[0].linlin(0, 0.005, [100, 101], 700);
    SinOsc.ar(mod, 0, 0.1)
}.play
```

Figure 3. Mass-Spring-Damper (MSD) used for FM.

See the help files of the classes Fb1_VanDerPol, Fb1_Duffing, Fb1_Hopf, Fb1_Lorenz, and Fb1_MSD for further examples. The adaptive variants Fb1_HopfA and Fb1_HopfAFDC can preserve the frequency of an external force after its stopping. That provides an unusual synthesis option. The classes implement the techniques described in [22] and [23], which are generic ways to make ODE systems adaptive.

2.3 Models from various fields

ODE models occur in many fields like physics (eminently mechanics), electrical engineering, population dynamics, and even chemistry. They can be interesting audio engines themselves or act as a starting point for further explorations. For an overview, see the SIAM publication *Exploring ODEs* [24] with an experimental approach in the graphic domain and many links and examples. Also, it's worth being aware that ODEs of one type can occur in different forms. The decision for a particular parameter set can have a vast impact on audio usability. Ultimately, parameter spaces demand a practical investigation. Fb1_ODE's help file contains examples from mechanics (driven pendulum, reduced two-body problem, Ex. 8a/b) and population dynamics (Lotka-Volterra and Hastings-Powell, Ex. 9a/b).

2.4 Modulations

It's possible to modulate systems parameters and the time scaling factor at audio rate – the latter can also get negative values. The example of Figure 4 varies that of Figure 3 by modulations of external force, mass, and time scaling (`tMul` argument). The specific choice preserves the development of the previous example – in general, it is easy to make systems unstable by operations of such kind. While the changing mass might still have a physical plausibility, the use of changing and even negative time steps is finally destroying a correct integration, though still possibly useful as a synthesis option.

```
x = {
    var f = SinOsc.ar(0.1).range(0.18, 0.2),
        mass = SinOsc.ar(1.2).range(1, 10) * 0.0001,
        spring = 50, dampen = 0.0005, sig, mod;
    sig = Fb1_MSD.ar(f, mass, spring, dampen,
        tMul: SinOsc.ar(5).range(-0.2, 1.5),
        leakDC: false
    );
    mod = sig[0].linlin(0, 0.005, [100, 101], 700);
    SinOsc.ar(mod, 0, 0.1)
}.play
```

Figure 4. MSD with modulations, used for FM.

2.5 Integration methods

While the use of symplectic integration procedures has advantages for the cited reasons, Fb1_ODE supports other families of integration methods like Euler, Prediction-Evaluation-Correction (PEC), Runge-Kutta, Adams-Bashforth, and Adams-Bashforth-Moulton as well. As an advanced feature, more integration methods can be added interactively with the class Fb1_ODEintdef. In some cases, alternative integration methods can lead to timbral variations. Quite often, though, they lead to blowups or decays, where stable oscillations should occur. Figure 5 shows an example with a 3rd order Runge-Kutta integration of the harmonic oscillator, which leads to decay after a few seconds.

```
// Needs Fb1_ODEdef from Fig. 1.
// Decay after a few seconds with Runge-Kutta 3rd order.
// ATTENTION: blowups with other non-symplectic procedures!
// Euler variants are particularly bad.

x = {
    var sig = Fb1_ODE.ar(\harmonic,
        tMul: 1000 * 2pi,
        intType: \rk3
    );
    Limiter.ar(sig) * 0.1 * EnvGen.ar(Env.asr(0.1))
}.play
```

Figure 5. Failing integration with 3rd order Runge-Kutta.

The symplectic "rattle" procedure has another advantage: it offers the option to improve accuracy by the iterated division of step sizes. To employ these variants, pass the SC Symbol 'sym' with one of the suffixes 2, 4, 6, 8, 12, 16, 32, 64 on Fb1_ODE's `intType` argument (default 'sym4').

2.6 Handling unstable systems

It is possible to insert an additional function, which applies to every array of samples that is the intermediate result – and next input – of the numerical integration procedure. Consequently, the correct integration of the ODE is out of scope. The option still has its value: limiting functions can prevent systems from blowing up. Let us regard this paraphrase on the Mass-Spring-Damper model:

$$m\,y''(t) = -k\,y(t) - c\,y'(t) + F(t) + y(t)\,y'(t) \qquad (9)$$

The use of this ODE in similar ways as in the examples from Figures 3-4 leads to a derail by infinite numbers after few seconds. However, the system remains inside practical bounds with a limiting operator like `clip2` (Figure 6).

```
Fb1_ODEdef(\MSD_ext, { arg t, y, f = 0,
    m = 1, s = 1, d = 0;
    [
        y[1],
        f - (d * y[1]) - (s * y[0]) + (y[0] * y[1]) / m
    ]
}, 0, [0, 0]);

x = {
    var f = 0.5, m = 0.001, s = 150, d = 0.001, mod;
    // smooth random LFO for clip values
    var lfo = LFDNoise3.kr(2).exprange(0.2, 1500);
    var sig = Fb1_ODE.ar(\MSD_ext,
        [f, m, s, d],
        leakDC: false,
        compose: { arg y; y.clip2(lfo) }
    );
    mod = sig[0].linlin(0, 0.006, [50, 50.1], 700);
    SinOsc.ar(mod, 0, 0.1)
}.play
```

Figure 6. MSD with disturbance, used for FM.

2.7 Further options and settings

Initial values – the system state y_0 at time t_0: $y(t_0) = y_0$ – are essential for an ODE solution. The user can pass them on Fb1_ODE with the corresponding arguments `t0` and `y0`. In many cases – like Mass-Spring-Damper with constant external force – a change of the start time does not have a consequence: the resulting waveform is the same, whereas the initial system values (position and velocity) have a significant impact.

Fb1_ODE can also return additional information in separate channels. By default, it returns the solution function(s) of the ODE system. Optionally, it can also output the differential and the time – which is not necessarily linear, as there might be a time modulation. The corresponding arguments are `withDiffChannels` and `withTimeChannel`. See the Fb1_ODE help file examples 6a and 6b.

Fb1_ODEdef allows for amplitude scaling factors. Usually, they default to 1, but certain ODEs, like Lorenz, produce a very high amplitude level with standard parameters. Therefore, it makes sense to scale their output down by default. However, with Fb1_ODE's `withOutScale` argument, the default scaling can be disabled (Fb1_ODE help file examples 7).

As many system solutions produce an unwanted DC offset, a DC leaker applies by default. The user can disable the option with Fb1_ODE's `leakDC` argument.

One of Fb1_ODE's basic arguments is `tMul` for time scaling. As in Figure 1, it can determine the resulting frequency (alternatively, the user might define the multiplication in the system definition with Fb1_ODEdef). It is important to note that numerical integration in the audio rate case is always performed on a per-sample base – even if the block size is larger than 1 – only the unit of the system time is varied. However, scaling is restricted to numerical accuracy limits: with extreme `tMul` values or numerically sensitive equations, you might encounter blowups or situations where the resulting frequency does not linearly relate to the scaling. Besides, Fb1_ODE can alternatively run at control rate (Fb1_ODE.kr), which helps to save CPU-load if the current block size is larger than 1.

2.8 Workflow recommendations, troubleshooting

The direct definition of the ODE systems in the language is a convenience that comes with the price of a possibly large number of unit generators involved. That does not necessarily mean a high CPU-load of the audio engine but leads to a higher compile-time. The user might want to extend SC's server resources before booting, e.g., set a higher number of unit generators with the server option `numWireBufs`. For a smooth workflow, I would recommend taking a reduced `blockSize` (e.g., 1, 2, 4, 8, 16) while experimenting because compile-time is shorter. But after fixing the design of a `SynthDef`, it might pay going back to a `blockSize` value of 32 or 64 for runtime efficiency, even more if many control rate unit generators are involved.

Especially with custom-designed ODEs, the usage of Fb1_ODE is – inherently – highly experimental. I strongly recommend being careful with amplitudes! Sudden blowups might result from the mathematical characteristics of the ODE systems. They might also stem from parameter adjustments – on which ODEs can react with extreme sensitivity – or from numerical accumulation effects. As a precautionary measure, users can employ SC's limiting/distorting operators (`tanh`, `clip`, `softclip`, `distort`) with the compose option (2.6) or external limiting, e.g., with the quarks JITLibExtensions (MasterFX) or SafetyNet.

The numerical integration procedure supposes well-defined ODE systems. The Fb1_ODE framework doesn't perform any checks concerning the principal existence and uniqueness of an ODE solution.

3. CONCLUSIONS

The SC class extension Fb1_ODE enables the audification of ordinary systems of differential equations with initial values in realtime. ODEs serve as a generic description system for waveforms, which one often cannot define by (explicit) mathematical means. Synthesis experiments have proven to be promising with well-known ODE systems from many scientific fields, as well as with custom-designed ODEs. Options for the modulation of ODE parameters and system time – and the integration mechanism itself – blur the model concept, though, also widen the field of sonic exploration.

Acknowledgments

Great thanks to my colleague David Pirrò of IEM Graz, who gave me a nudge to dive into the world of ODE audification, and his recommendation of symplectic integration procedures. Fb1_ODE rests upon the Fb1 class, therefore also big thanks to Nathaniel Virgo, whose idea for single sample feedback – with block sizes greater than 1 – lives therein.

4. REFERENCES

[1] D. Mayer, miSCellaneous_lib, SuperCollider quark extension: https://github.com/dkmayer/miscellaneous_lib, Accessed February 20, 2021.

[2] S. Wilson, D. Cottle and N. Collins (Eds.), The SuperCollider Book. The MIT Press, Cambridge, 2008.

[3] SuperCollider page, https://supercollider.github.io, Accessed February 20, 2021.

[4] D. Mayer, "Fb1 – an interface for single sample feedback and feedforward in SuperCollider," in Proc. Int. Conf. Sound and Music Computing (SMC2020), Torino, 2020, pp. 200–203. https://smc2020torino.it/adminupload/file/SMCCIM_2020_paper_53.pdf, Accessed February 20, 2021.

[5] S. Bilbao, Numerical sound synthesis: Finite difference schemes and simulation in musical acoustics, J. Wiley, Chichester, UK, 2009.

[6] D. Slater, "Chaotic sound synthesis," in Computer Music Journal, vol. 22, no. 2, pp. 1219, 1998.

[7] I. Choi, "Interactive exploration of a chaotic oscillator for generating musical signals in realtime concert performance," in Journal of the Franklin Institute, vol. 331, no. 6, pp. 785–818, 1994.

[8] X. Rodet, "Sound and Music from Chua's Circuit," in Journal of Circuits, Systems and Computers, vol. 3, no. 1, pp 49–61, 1993.

[9] B. Jacobs, "A Differential Equation Based Approach to Sound Synthesis and Sequencing," in International Computer Music Conference Proceedings, pp. 557–561, 2016.

[10] A. Falaize, T. Hélie, "Passive guaranteed simulation of analog audio circuits: A port-Hamiltonian approach," in Applied Sciences, vol. 6, no. 10, p. 273, 2016.

[11] M. Holters and U. Zölzer, "A generalized method for the derivation of non-linear state-space models from circuit schematics," Proc. 23rd European Signal Process. Conf. (EUSIPCO 2015), (Nice, France), pp. 1073–1077, IEEE, August 2015.

[12] K. Dempwolf, M. Holters, and U. Zölzer, "Discretization of Parametric Analog Circuits for Real-Time Simulations," Proc. of the 13th Int. Conference on Digital Audio Effects (DAFx-10), Graz, Austria, September 6-10, 2010, pp. 42–49.

[13] A. Fettweis, "Wave Digital Filters: Theory and Practice," Proc. IEEE, Vol. 74, No. 2, pp. 270–327, Feb. 1986.

[14] N. Stefanakis, M. Abel, and A. Bergner, "Sound synthesis based on ordinary differential equations," in Computer Music Journal, vol. 39, no. 3, pp. 46–58, 2015.

[15] A. Bernardini, P. Maffezzoni, and A. Sarti, "Linear Multistep Discretization Methods With Variable Step-Size in Nonlinear Wave Digital Structures for Virtual Analog Modeling," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 27, no. 11, pp. 1763-1776, Nov. 2019.

[16] E. Hairer, C. Lubich, and G. Wanner, Geometric numerical integration: structure-preserving algorithms for ordinary differential equations, Springer, 2006.

[17] D. Pirrò, Composing Interactions. Dissertation, Institute of Electronic Music and Acoustics, University of Music and Performing Arts Graz, 2017. https://pirro.mur.at/media/pirro_david_composing_interactions_print.pdf, Accessed February 20, 2021.

[18] D. Pirrò, Henri – llvm-based sound synthesis and dynamical systems numerical integration environment. https://git.iem.at/davidpirro/henri, Accessed February 20, 2021.

[19] G. Strang, "On the Construction and Comparison of Difference Schemes," in SIAM Journal on Numerical Analysis, Vol. 5, No. 3 (Sep., 1968), pp. 506-517.

[20] R.I. McLachlan, G.R.W. Quispel, "Splitting methods," in Acta Numerica, Volume 11, January 2002, pp. 341–434.

[21] F.G. Germain, "Fixed-rate modeling of audio lumped systems: A comparison between trapezoidal and implicit midpoint methods," in Proc. 20th Int. Conf. Digital Audio Effects (DAFx-17), Edinburgh, UK, Sept. 2017, pp. 168–175.

[22] L. Righetti, J. Buchli, and A.J. Ijspeert, "Adaptive Frequency Oscillators and Applications," in The Open Cybernetics and Systemics Journal, 3, pp. 64–69, 2009. https://www.researchgate.net/publication/41666931, Accessed February 20, 2021.

[23] T. Nachstedt, C. Tetzlaff, and P. Manoonpong, "Fast Dynamical Coupling Enhances Frequency Adaptation of Oscillators for Robotic Locomotion Control," in Frontiers in Neurorobotics. Published online March 21, 2017: https://www.frontiersin.org/articles/10.3389/fnbot.2017.00014/full, Accessed February 20, 2021.

[24] L. Trefethen, A. Birkisson, and T.Driscoll, Exploring ODEs. SIAM – Society for Industrial and Applied Mathematics, 2018. http://people.maths.ox.ac.uk/trefethen/Exploring.pdf Accessed February 20, 2021.

SIGNAL REPRESENTATIONS FOR SYNTHESIZING AUDIO TEXTURES WITH GENERATIVE ADVERSARIAL NETWORKS

Chitralekha GUPTA (chitralekha@nus.edu.sg) (0000-0003-1350-9095)[1],
Purnima KAMATH (purnima.kamath@u.nus.edu) (0000-0003-0351-6574)[1], and
Lonce WYSE (lonce.wyse@nus.edu.sg) (0000-0002-9200-1048)[1]

[1]**National University of Singapore**, Singapore

ABSTRACT

Generative Adversarial Networks (GANs) currently achieve the state-of-the-art sound synthesis quality for pitched musical instruments using a 2-channel spectrogram representation consisting of log magnitude and instantaneous frequency (the "IFSpectrogram"). Many other synthesis systems use representations derived from the magnitude spectra, and then depend on a backend component to invert the output magnitude spectrograms that generally result in audible artefacts associated with the inversion process. However, for signals that have closely-spaced frequency components such as non-pitched and other noisy sounds, training the GAN on the 2-channel IFSpectrogram representation offers no advantage over the magnitude spectra based representations. In this paper, we propose that training GANs on single-channel magnitude spectra, and using the Phase Gradient Heap Integration (PGHI) inversion algorithm is a better comprehensive approach for audio synthesis modeling of diverse signals that include pitched, non-pitched, and dynamically complex sounds. We show that this method produces higher-quality output for wideband and noisy sounds, such as pops and chirps, compared to using the IFSpectrogram. Furthermore, the sound quality for pitched sounds is comparable to using the IFSpectrogram, even while using a simpler representation with half the memory requirements.

1. INTRODUCTION

In recent years, GANs have achieved the state-of-the-art performance in neural audio synthesis, specifically for pitched musical instrument sounds [1, 2]. Engel et al. [1] showed that a progressively growing GAN [3] can outperform strong WaveNet [4] and WaveGAN [5] baselines in the task of conditional musical instrument audio generation achieving comparable audio synthesis quality and faster generation time. Nistal et al. [2] further showed that a 2-channel input representation consisting of the magnitude and the instantaneous frequency (IF) of the Short-Time Fourier Transform (STFT) achieves the best synthesis results in this framework compared to other kinds of representations, such as Mel spectrogram, MFCC, and Constant-Q Transform. The derivative of unwrapped phase of a signal with respect to time is equal to the angular difference between the frame stride and signal periodicity, and is commonly referred to as the instantaneous frequency (IF). Estimation of IF provides comprehensive information about the phase of the signal when the audio is pitched, i.e. has components that are clearly separated in frequency. Thus, a magnitude spectrogram combined with the estimated IF results in high-quality reconstruction of the signal for pitched signals such as musical instruments. In broadband and noisy short duration signals, components are not separated in frequency, and neighboring frequency bins have complex and highly interdependent amplitude and phase relationships that are necessary for reconstruction and the representation is very sensitive to IF estimation errors.

DrumGAN [6] extended the work in [2] to various drum sounds, however the authors have notably not used the IF spectrogram that produce state-of-the-art quality for pitched sounds, but instead, use spectrograms of the real and imaginary parts from the STFT directly. They also use a set of perceptually correlated features more appropriate than pitch for conditioning the percussion sounds in the target data set.

Průša et al. [7] proposed a non-iterative phase reconstruction algorithm called Phase Gradient Heap Integration (PGHI) that uses the mathematical relationship between the magnitude of Gaussian windowed STFT and the phase derivatives in time and frequency of the Fourier transform to reconstruct the phase using only the magnitude spectrogram. Marafioti et al. [8] compared three different GAN architectures, and showed that for a dataset consisting of spoken digits and piano music, the architecture using PGHI produced audio of objectively and perceptually higher quality than the other representations they compared based on an aggregate set of different signal types. A direct comparison with GanSynth [1] which was being published at about the same time was also not included in their study.

In this paper, we study and compare the state-of-the-art GanSynth with magnitude spectrogram+IF audio representation and reconstruction method and the PGHI method of representation and reconstruction for a systematically organized collection of audio textures such as pitched musical instruments, noisy pops, and chirps, spanning a range from pitched steady-state to broadband signals. We show

that the PGHI method of reconstruction from GAN estimates is more robust for synthetic spectrograms and estimation errors for different kinds of input signals than the state-of-the-art magnitude+IF representation. This study contributes to the development of general and efficient representations for training GANs for complex audio texture synthesis.

2. AUDIO TEXTURES AND REPRESENTATIONS

2.1 Audio Representations and Inversion Techniques

Many algorithms learn to estimate the magnitude spectrogram and then use iterative methods such as Griffin-Lim [9] to estimate the phase and reconstruct the time domain signal. However, these traditional methods of phase estimation and reconstruction are known to have perceptible artifacts in the reconstructed signal. Estimation of phase is difficult and prone to errors in part because artificial or manipulated images may not produce a real-valued time domain signal when inverted.

Another way of representing phase is with instantaneous frequency. The estimate of magnitude spectrogram and IF in frequency domain can be used to reconstruct a time domain signal by computing the unwrapped phase from the cumulative sum of IF across time axis, and computing an inverse Fourier transform. The state-of-the-art GANSynth framework [1,2] estimates this 2-channel audio representation, i.e. log magnitude and IF, or IFSpectrogram. Engel et al. hypothesized and showed that synthesized audio quality from the IFSpectrogram is robust to estimation errors for the NSynth dataset of pitched musical instrument audio while noting the importance of choosing analysis window sizes large enough to be primarily sensitive to a single frequency component. However, to the best of our knowledge, IFSpectrogram method has not been tested and compared to other representations for non-pitched and noisy sounds.

We observe that whether converting pitched instrument or noisy transient audio into IFSpectrogram representation, that resynthesizing produces a high quality audio output for both the kinds of sounds. However, if we add a small Gaussian noise to the IF channel (to simulate estimation error in IF) and then resynthesize, the perceptual quality of the pitched sounds is not affected as much as the quality of the noisy pop sounds. Audio examples of this simulation are presented in the companion website [1]. This indicates that IFSpectrogram method may not be robust to manipulated and synthetic spectrograms or estimation errors for non-pitched and noisy sounds.

For a signal composed of sinusoidal components with constant frequencies, the phase grows linearly in time for all the frequency channels that have energy in the spectrogram. For these frequency channels, the IF is constant and the local group delay (STFT phase derivative with respect to frequency) is zero. However, in case of an impulse train, the situation is reverse to that of sinusoidal components, wherein the phase derivative with respect to frequency axis will have more information than the IF as there is energy across almost all the frequency channels in the spectrogram, but the change of phase with respect to time exists only around the impulse events, and otherwise it is zero. Furthermore, for signals that have fast moving or closely spaced frequency components, IF does not capture the variability in the frequency direction.

The Phase Gradient Heap Integration (PGHI) method [7] is a non-iterative phase estimation method that exploits the mathematical relationship between the time and frequency derivatives of log magnitude spectrogram with the phase gradients in frequency and time axes respectively. To provide a brief summary here, Průša et al. [7] proved mathematically and experimentally that the derivative of phase along frequency axis $\phi_\omega(m,n)$ and, the derivative of phase along time axis $\phi_t(m,n)$ can be estimated solely from the time and frequency derivatives of log-magnitude of STFT ($\mathrm{s}_{\log_t}$, $\mathrm{s}_{\log_\omega}$) respectively computed with a Gaussian window, as [10, 11],

$$\begin{aligned}\phi_\omega(m,n) &= \frac{-\gamma}{2aM}(\mathrm{s}_{\log_t}(m,n)) \\ \phi_t(m,n) &= \frac{aM}{2\gamma}(\mathrm{s}_{\log_\omega}(m,n)) + 2\pi am/M\end{aligned} \tag{1}$$

where, M is the number of frequency channels, a is the hop size, and γ is the time-frequency ratio of Gaussian window, which is recommended to be aM/L, L being the length of the input signal in samples. Although the theory behind the non-iterative method of phase reconstruction from the STFT magnitude holds for Gaussian continuous window, Prusa et al [7] showed that the algorithm works well for a discretised truncated Gaussian window, however with the Gaussian approximation of other windows such as Hann and Hamming windows, they found significant signal degradation. Therefore in this work, we have used the truncated Gaussian window function. Redundancy between frames should be such that there is sufficient dependency between the values of the STFT to facilitate magnitude-only reconstruction. The recommended redundancy is $M/a \geq 4$ [8].

This method also implements a numerical integration of these phase gradients such that integration is first performed along the prominent contours of the spectrogram in order to reduce accumulation of the error, and so on. This heap integration method to estimate phase from the phase gradients helped to make the synthesis robust to estimation errors and noise [7, 10].

In this work, our goal is to investigate the quality of audio produced by a progressive GAN trained on a single channel log magnitude spectrogram and using PGHI for inversion of the estimated spectrogram to time domain signal and compare it to using the two-channel IFSpectrogram representation, for wideband, noisy, non-pitched or fast changing signals, as well as pitched instrument signals. With this framework, we propose a general approach for audio synthesis using the state-of-the-art GAN that works for a variety of different sounds.

[1] https://animatedsound.com/amt/listening_test_samples/#simulation

2.2 Audio Textures

Audio synthesis finds practical applications in creative sound design for music, film, and gaming, where creators are looking for sound effects suited to specific scenarios. Research in this field aims to learn a compact latent space of audio such that adjustments to these latent variables would help the creator search through a known space of sounds (eg. water drops and footsteps), parametrically control (eg. rate of water dripping) as well as explore new sounds in the spaces in between the known sounds [5].

Building upon generative adversarial image synthesis techniques, researchers exploring GAN techniques for neural audio synthesis have made significant progress in building frameworks for conditional as well as unconditional synthesis of a wide range of musical instrument timbres [1, 2]. These models are trained on NSynth dataset [12] that consists of notes from musical instruments across a range pitches, timbres, and volumes. Conditioning on pitch allows the network to learn natural timbre variation while providing musical control of notes for synthesis. The NSynth dataset provides a comprehensive representation of pitched sounds comprised primarily of well-separated harmonics. There has been some work on audio texture modeling for synthesis [13–15] including deep learning approaches [16], but audio textures have received considerably less attention than traditional musical sounds and speech.

Sound textures [13, 17] have more timbral variation including wideband or noisy components, such as footsteps or motors, and a wide range of temporal structure not found in pitched instruments. Furthermore, there can be very fast-varying frequency components and pitches in sounds such as water dripping, and chirps. Thus we examine the performance of controlled audio synthesis techniques on trained networks using three types of sounds - pitched instruments, noise burst pops, and frequency sweep chirps, as shown in Figure 1. In this work, we conduct experiments on pitched musical instruments and carefully controlled synthetic non-pitched and dynamic textures. More complex and natural textures are left for future study.

2.3 Conditional GAN architecture for audio synthesis

Parametrically controllable audio synthesis has also been an active field of research in recent years. Hsu et al. [18] used hierarchical variational autoencoders (VAEs) for conditional or controlled speech generation. Similarly, Luo et al. [19] learn separate latent distributions using VAEs to control the pitch and timbre of musical instrument sounds. Engel et al. [12] conditioned a WaveNet-style autoregressive model to generate musical sounds, as well as interpolate between sounds to generate new sounds. The current state-of-the-art performance in conditional synthesis of audio is the GANSynth architecture [1] which introduces a progressively growing Wasserstein GAN for controlled music synthesis and is based on the IFSpectrogram representation [2]. Thus, we adopt this architecture with IFSpectrogram representation as our baseline.

3. EXPERIMENTAL DETAILS

3.1 Audio Datasets

3.1.1 Pitched Musical Instruments

We make use of the NSynth dataset [12], that consists of approximately 300,000 single-note audios played by more than 1,000 different instruments. It contains labels for pitch, velocity, instrument type, acoustic qualities (acoustic or electronic), and more, although, for this particular work, we only make use of the pitch information as the conditional parameter. We use the same subset of this dataset as was used by Nistal et al. [2]. It contains acoustic instruments from the brass, flutes, guitars, keyboards, and mallets families, and the audio samples are trimmed from 4 to 1 seconds and only consider samples with a MIDI pitch range from 44 to 70 (103.83 - 466.16 Hz). This yields a subset of approximately 22,000 audio files with balanced instrument class distribution.

3.1.2 Noisy Pops

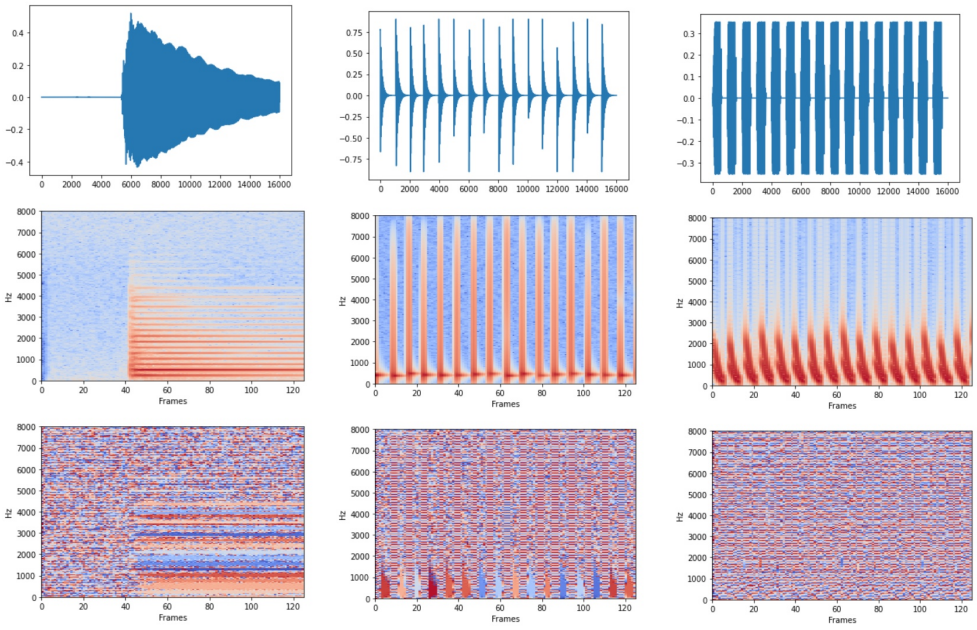
On the other end of the spectrum of sounds we tested are *pops*. A pop is a burst of noise filtered by a bandpass filter. We generated the pop textures with three parameters - rate (number of events per seconds), irregularity in the temporal distribution (using a Gaussian distribution around each evenly-spaced time value), and the center frequency of the bandpass filter. Rate ranges from 2 to 16 pops per second, center frequency ranges from 440 to 880 Hz (corresponding to midi pitch values 69 to 81), and irregularity described by a Gaussian distribution with a standard deviation ranging from 0.04 to 0.4. We generate 21 values for each of these three parameters, and five one-second long audio clips of each combination, resulting in a total of 46,305 ($21 \times 21 \times 21 \times 5$) audio files.

3.1.3 Chirps

In between the quality of the pitched sounds with relatively steady frequency components and the noisy pop sounds with sharp broadband transients are *chirps*. A chirp is a signal in which the frequency increases or decreases quickly with time. The chirps were generated with two frequency components space by an octave, and were controlled with 5 parameters - irregularity in time (like the pops), chirp rate (2 to 16 chirps per second, 9 samples), frequency sweep range in octaves indicating steepness of chirp ([-3. -1, 1, 3] where negative is descending and positive is ascending), event duration i.e. duration of each chirp in seconds (5 linearly spaced samples in [.02, .2]), and center frequency (9 linearly space samples in musical pitch space between 440 and 880 Hz). We generate 5 variations of each parameter (different due to the statistical distribution of events in time) resulting in a total of 40,500 ($5 \times 9 \times 4 \times 5 \times 9 \times 5$) audio files of 1 second each.

3.2 GAN architecture

We used the progressively growing Wasserstein GAN architecture [1, 2] which consists of a generator G and a discriminator D, where the input to G is a random vector z



(a) Pitched Instrument (Piano) (b) Pops (c) Chirps

Figure 1. Examples of (a) a pitch instrument (piano), (b) Noise burst or pops, and (c) Frequency sweeps or chirps, with their respective audio waveform (top row), log magnitude spectrogram (middle row), and instantaneous frequency of unwrapped phase (bottom row) plots. The audio examples presented are 1 second long at 16 kHz sampling rate. Spectrogram computation is with window size 512 and hop size 128 samples.

with 128 components from a spherical Gaussian distribution along with a one-hot conditional vector c_{in}. Separate models were trained for each data set with the only difference being the dimension of the one-hot pitch vector (27, 13, and 9 for NSynth, pops, and chirps, resp.) For each dataset, we train two models as shown in Figure 2. Model A uses a 2-channel audio representation consisting of the log magnitude spectrogram and IF (Figure 2(a)) computed from Short Time Fourier Transform (STFT) with Hanning window, and Model B uses a single-channel log magnitude of Gabor transform (i.e. STFT with Gaussian window) audio representation (Figure 2(b)). During generation, Model A's estimated IFSpectrogram is inverted to a real time domain signal using Librosa's inverse STFT which uses Griffin-Lim iterative algorithm for synthesis initialized by the estimated phase from IF. For model B, we use phase gradient heap integration (PGHI) [7] [2] for reconstruction of the audio signal from the log magnitude. It reconstructs the phase only for the positive frequency coefficients and enforces conjugate symmetry to the negative frequency coefficients in order to guarantee a real-valued time domain signal.

The generator's architecture consists of a Format block and a stack of Scale blocks. The Format block turns the 1D input vector z + one-hot conditional c_{in}, with 128 + x dimensions (where x could be 27, 13, or 9) into a 4D convolutional input consisting of [batch size, 128, w_0, h_0], where w_0 and h_0 are the sizes of each dimension at the

[2] https://github.com/andimarafioti/tifresi

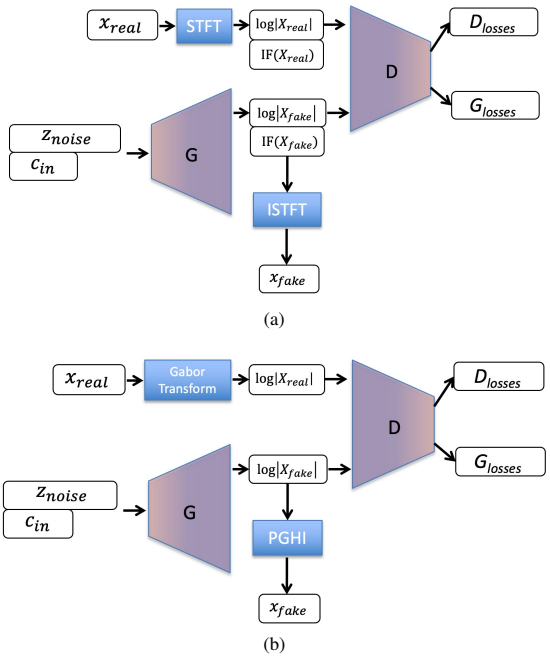


Figure 2. GAN block diagram with (a) IF, and (b) PGHI. z_{noise} is the 128 dimensional latent vector, c_{in} is the conditional parameter one-hot vector. G is the generator, D is the discriminator.

input of the scale block.

The scale blocks are a stack of convolutional and box-upsampling blocks that transform the convolutional input to the generated output signal progressively in 5 stages. The discriminator D is composed of convolutional and down sampling blocks, mirroring the configuration of the generator. D estimates the Wasserstein distance between the real and generated distributions. For more details, please refer to [2] [3] . Our code that implements the GAN architecture with IF as well as PGHI methods (an extended version of Nistal et al.'s code) is available here [4] .

3.2.1 Training

Training is divided into 5 stages, wherein each stage a new layer, generating a higher-resolution output, is added to the existing stack, which is the essence of the progressive-GAN [1, 3]. The gradual blending in of the new layers with a blending parameter *alpha* ensures minimum possible perturbation effects as well as stable training. We train all the models for 1.2M iterations on batches of 8 samples: 200k iterations in each of the first three phases and 300k in the last two. Adam optimization method is employed.

We tried multiple FFT sizes 512, 1024, and 2048 to compute the time-frequency representations, that correspond to window sizes of 32 ms, 64 ms, and 128 ms respectively for a signal sampled at 16 kHz. For transient sounds, a window size with higher time resolution is needed, i.e. a shorter window and indeed we empirically found that FFT size of 512 serves well for both the transient pop sounds and the steady pitched sounds. Therefore, in the experiments presented in this paper, we use FFT size of 512. We tested the effect of redundancy between frames in reconstruction, thus we trained two models, with hop sizes 64 and 128, i.e. 87.5% and 75% overlap between consecutive frames. We train two types of models IF and PGHI, for three kinds of audio textures, NSynth, pop, and chirp, for each of the two hop sizes. All of the models took about 2.5 to 3 days to train on an Nvidia Tesla V100-32GB GPU.

3.3 Evaluation Metrics

Evaluation of generative models is challenging, especially when the goal is to generate perceptually realistic audio that may not be exactly same as any real audio in the dataset. Previously, the inception score has been used as the objective measure that evaluates the performance of a model for a classification task such as pitch or instrument inception score [1, 2]. However, in this work, we are comparing signal representations and synthesis techniques, while the GAN architecture remains the same. Since the variety of sounds with respect to classification is not expected to change. Indeed, Nistal et al [2] noted that inception models are not robust to the particular artifacts of the representations they were comparing, and therefore, it is not a very reliable measure of the overall generation quality.

Marafioti et al. [8] developed an interesting *consistency* measure that estimates how close a magnitude spectrogram is to the frequency transform of a real audio signal. However, it is not obvious how it could be used to compare representations that include explicit phase representations. Also, the perceptual quality of the generated audio signal depends on other factors as well. For example, a real-valued time domain signal of poor perceptual quality will have a perfectly consistent magnitude spectrogram.

In this work, we performed listening tests for subjectively evaluating the quality of the generated sounds, as well as computed Fréchet Audio Distance (FAD) [20] as the objective evaluation metric.

3.3.1 Human Evaluation

To construct stimuli for listening experiments, three points in the latent space are randomly chosen to generate three audio signals of 1 second each per pitch class per trained model, which were then stitched together with a 0.5 second silence before each of the 3 segments) resulting in a 4.5 seconds duration audio clips that were presented in the listening test. This provided variability within each clip so that the listeners focus on the sound quality of the clips and not on the instrument type or the rate of pops and chirps. For reference, a similar set of audio clips was prepared from the original or real audio data set as well.

The listening test was conducted by recruiting twenty participants via Amazon's Mechanical Turk (AMT) website. In each assessment task, the participants were asked to listen first to the reference, then to the two synthesized audio clips, randomly ordered, and then to select the one they felt was the closest in sound quality to the reference clip, or if they were similar. Our task instructions were simplified for the participants and included text like "Although the synthetic clips may sound quite different from the original, you will need to select a clip whose sound quality is most similar to the sound quality of the original". The two audio clips belonged to either IF or PGHI reconstruction techniques for a hop size of 64 or 128 for each comparison. Only same type of sounds were compared, i.e. NSynth_IF to NSynth_PGHI, pop_IF to pop_PGHI etc. Moreover, the two clips being compared had the same pitch or center frequency. 20 random pitches from the NSynth dataset, 13 pitches from pops, and 9 pitches from chirps were selected to build a sample size of 84 comparison trials (42 comparisons each for hop 64 and 128 reconstructions respectively) and overall 1,680 ratings were collected. The trials were loaded into AMT in a random sequence and were completed by participants within 2 hours. The participants were compensated at the rate of US$ 0.02 per comparison trial.

3.3.2 Fréchet Audio Distance

The Frechet Audio Distance (FAD) [20] [5] is the distance between the statistics (mean and covariance) of real and fake data computed from an embedding layer of the pre-trained VGGish model. The embedding layer is considered

[3] https://github.com/SonyCSLParis/Comparing-Representations-for-Audio-Synthesis-using-GANs

[4] https://github.com/lonce/sonyGanFork

[5] https://github.com/google-research/google-research/tree/master/frechet_audio_distance

to be a continuous multivariate Gaussian, where the mean and covariance are estimated for real and fake data, and the FAD between these is calculated as:

$$FAD = ||\mu_r - \mu_g||^2 + tr(\Sigma_r + \Sigma_g - 2\sqrt{\Sigma_r \Sigma_g}) \quad (2)$$

where μ_r, Σ_r and μ_g, Σ_g are the mean and covariances of real and fake probability distributions, respectively. Lower FAD means smaller distances between synthetic and real data distributions. The VGGish model is trained on 8M Youtube music videos with 3K classes. The FAD metric has been tested successfully specifically for the purpose of reference-free evaluation metric for enhancement algorithms. FAD performs well in terms of robustness against noise, computational efficiency, and consistency with human judgments, and has been used by Nistal et al. [2]. FAD has been found to have a high correlation (0.52) with human perceptual judgment compared to other measures such as signal-to-distortion ratio, cosine distance or magnitude-L2 distance [20].

4. RESULTS AND DISCUSSION

Qualitatively it is observed that with the IF method, the sharp transients of the pop sounds get smeared in time, whereas PGHI method produces clear and sharp transients. This temporal smearing effect is also observed in the short duration chirps generated from the IF method. This smearing effect arises from the inability of IF to provide robust information about phase when the signal contains closely spaced wideband frequency components. For NSynth data, however, the two methods sounded approximately equal in quality. Examples of the synthesised audio presented for listening tests are here [6], and visual analysis of the generated spectrograms are provided here [7].

Figure 3 (a) and (b) show results from the listening test for reconstructions using hop sizes 64 and 128 respectively. For both hop sizes, participants rated PGHI reconstructions to be significantly better than IF for pop sounds, where they rated in favour of PGHI 80.79% and 73.15% for hop sizes 128 and 64 respectively. This result clearly shows that PGHI with GAN produces perceptually higher quality audio for noisy signals. For chirp sounds, participants rated PGHI somewhat better than IF. But for NSynth pitched instrument sounds, PGHI and IF are similarly rated for both hop lengths. Furthermore, we observe that hop size 64 shows a clearer distinction in preference between IF and PGHI for nsynth and chirp sounds, than hop size 128. This indicates that a higher redundancy in the spectrogram representation may help in better reconstruction with PGHI method than IF method. However, comparison between the two hop sizes for the same method has shown mixed responses for the different datasets, which means that redundancy of more than 4 may not have a significant impact on the reconstructed audio quality of one method.This systematic study suggests that PGHI with GAN produces audio quality perceived as roughly equal to the state-of-the-art IF method for pitched sounds, but significantly higher as the complexity of the signal increases.

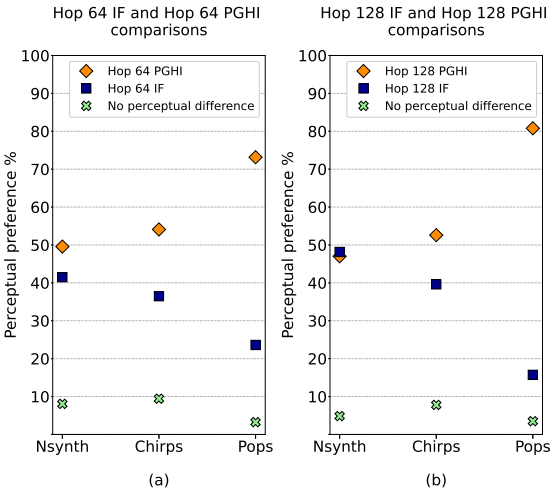


Figure 3. Results from listening tests for comparing IF and PGHI reconstructions from GAN using hop lengths of (a) 64 and (b) 128 respectively. Across both hop lengths, PGHI reconstructions of noise bursts or pops were rated to be significantly better than IF. For chirps, PGHI reconstructions were rated to be slightly better than IF and for pitched instruments PGHI reconstructions were rated almost similar to IF.

To evaluate objectively, we computed the FAD metric, as shown in Table 1. We observe that PGHI method generated audio that consistently shows a smaller distance from reference audio compared to that generated from IF method, although unlike the perceptual ratings, the two representations are closer for chirps than the other two signal types. While this objective measure is broadly in line with the higher ratings for the PGHI method, the systematic disagreement between the user and objective measures across pitched and chirp sounds demonstrate that there is more work to be done to find an objective measure that correlates with human judgements of quality.

The performance of the system, given all other settings are the same (training steps, architecture, etc), is better using the PGHI method than the IFSpectrogram method. Convergence during learning, especially in stage 5 of the progressive GAN differed between the two representations depending on the signal. The IF method representation converged better for NSynth, while PGHI representation converged slightly better for the other signals. However, in all cases, the quality of the synthesised audio was better (Figure 3) using the PGHI method.

5. CONCLUSIONS

We present a general method of audio synthesis using GAN that produces high quality audio output for a wide variety of sounds, pitched instruments as well as non-pitched

[6] https://animatedsound.com/amt/listening_test_samples/#examples
[7] https://animatedsound.com/amt/listening_test_samples/#analysis

Audio Texture	Hop Size	IF	PGHI
Pitched Instruments	128	1.500	1.001
Pitched Instruments	64	1.583	0.924
Pops	128	1.783	0.305
Pops	64	1.866	0.295
Chirps	128	1.395	1.031
Chirps	64	1.269	0.747

Table 1. FAD results of different GAN models with IF and PGHI. A lower FAD means smaller distances between synthetic and real data distributions.

and noisy pop and chirp sounds. We show that IFSpectrogram representation that currently produces the state-of-the-art performance with GAN for pitched instruments is not a robust representation for non-pitched and noisy sounds. Moreover, through subjective and objective measures, we show that integrating the PGHI representation and reconstruction technique in the GAN framework provides a reasonable solution to this problem, as it generates better audio quality for noisy pops and chirps than when using the IFSpectrogram method, and produces similar audio quality for pitched instruments. Audio examples generated from our experiments are available here [8] , and our code implementation is available here [9] .

A potential direction of improvement of the PGHI technique is to use the phase estimates from PGHI as a *warm-start* for other iterative phase reconstruction algorithms such as LeGLA, as shown by Prusa et al. [7]. Another possibility is to include different explicit representations of phase information in training that might outperform magnitude-only reconstruction with PGHI. Marafioti [8] used a representation with frequency derivatives for training which did not perform as well as the magnitude PGHI reconstruction method, but indicates the potential that this direction has to offer.

The method of training a GAN as a data-driven approach to designing parametrically controlled synthesizers holds a lot of promise for creative applications such sound design and music. A signal-independent representation for training the networks is an important step towards the universality and usability of this approach.

Acknowledgments

This research is supported by a Singapore MOE Tier 2 grant MOE2018-T2-2-127, and by an NVIDIA Corporation Academic Programs GPU equipment grant.

6. REFERENCES

[1] J. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts, "Gansynth: Adversarial neural audio synthesis," *arXiv preprint arXiv:1902.08710*, 2019.

[2] J. Nistal, S. Lattner, and G. Richard, "Comparing representations for audio synthesis using generative adversarial networks," in *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 161–165.

[3] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[4] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[5] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," *arXiv preprint arXiv:1802.04208*, 2018.

[6] J. Nistal, S. Lattner, and G. Richard, "Drumgan: Synthesis of drum sounds with timbral feature conditioning using generative adversarial networks," *arXiv preprint arXiv:2008.12073*, 2020.

[7] Z. Průša, P. Balazs, and P. L. Søndergaard, "A noniterative method for reconstruction of phase from stft magnitude," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 5, pp. 1154–1164, 2017.

[8] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "Adversarial generation of time-frequency features with application in audio synthesis," in *International Conference on Machine Learning*. PMLR, 2019, pp. 4352–4362.

[9] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 32, no. 2, pp. 236–243, 1984.

[10] Z. Průša, "The phase retrieval toolbox," in *Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio*. Audio Engineering Society, 2017.

[11] Z. Průša and P. Rajmic, "Toward high-quality real-time signal reconstruction from stft magnitude," *IEEE Signal Processing Letters*, vol. 24, no. 6, pp. 892–896, 2017.

[12] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1068–1077.

[13] N. Saint-Arnaud and K. Popat, "Analysis and synthesis of sound textures," in *in Readings in Computational Auditory Scene Analysis*. Citeseer, 1995.

[14] D. Schwarz, "State of the art in sound texture synthesis," in *Digital audio effects (DAFx)*, 2011, pp. 221–232.

[8] https://animatedsound.com/amt/listening_test_samples/

[9] https://github.com/lonce/sonyGanFork

[15] J. H. McDermott, A. J. Oxenham, and E. P. Simoncelli, "Sound texture synthesis via filter statistics," in *2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, 2009, pp. 297–300.

[16] J. M. Antognini, M. Hoffman, and R. J. Weiss, "Audio texture synthesis with random neural networks: Improving diversity and quality," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3587–3591.

[17] L. Wyse and M. Huzaifah, "Deep learning models for generating audio textures," in *Proceedings of the 2020 Joint Conference on Music Creativity*, Stockholm, Sweden, October 19-23 2020.

[18] W.-N. Hsu, Y. Zhang, R. J. Weiss, H. Zen, Y. Wu, Y. Wang, Y. Cao, Y. Jia, Z. Chen, J. Shen *et al.*, "Hierarchical generative modeling for controllable speech synthesis," *International Conference on Learning Representations (ICLR)*, 2019.

[19] Y.-J. Luo, K. Agres, and D. Herremans, "Learning disentangled representations of timbre and pitch for musical instrument sounds using gaussian mixture variational autoencoders," *International Society of Music Information Retrieval (ISMIR)*, 2019.

[20] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, "Fréchet audio distance: A metric for evaluating music enhancement algorithms," *arXiv preprint arXiv:1812.08466*, 2018.

CONSTRAINED DIFFERENTIAL EQUATIONS AS COMPLEX SOUND GENERATORS

Dario SANFILIPPO[1]

[1]**Independent**, Catania, Italy

ABSTRACT

This paper presents investigations for complex sound generation based on modified classic chaotic differential equations. The modification consists of a constraining structure, a cascaded saturating nonlinearity and DC-blocker unit within the recursive paths of the differential equations. The new configuration allows for the exploration of these equations in unstable regions, increasing the parameters ranges allowed and extending sonic possibilities. Furthermore, the configuration implements a low-level competing mechanism between positive and negative feedback relations that forces oscillation and enhances complex variations both at timbral and formal time scales. The resulting systems exhibit phase spaces with nontrivial trajectories and show potential for computer music and, particularly, applications requiring some degree of autonomy. The sound generators are implemented in the Faust programming language and are published on Github under the GNU GPL v3.0 license.

1. INTRODUCTION

Differential equations exhibiting chaotic behaviours have been studied for over a century and have found applications in several fields ranging from biology and chemistry to electronics and mechanics. The Lotka-Volterra equations are a system of two differential equations conceived in the 1910s and used to this day to model biological systems, particularly predator-prey interactions and dynamics [1]. The Duffing equation is a second-order differential equation and was initially investigated in the late 1910s to render the behaviour of particular damped oscillators [2]. In the 1920s, Balthasar van der Pol discovered a differential equation for a type of stable oscillator connected to behaviours in electrical circuits applying vacuum tubes [3]. The Lorenz system is a system of three differential equations developed in the 1960s as a mathematical model for atmospheric convection; the system later became one of the most iconic models displaying deterministic chaos [4]. Later, in the 1970s, Otto Rössler designed a chaotic attractor initially intended to behave similarly to the Lorenz attractor that was later found to be related to equilibrium in chemical reactions [5]. The Chua's circuit, developed in the 1980s, is an electrical circuit exhibiting chaotic behaviours that can be modelled through a system of three differential equations and that became famous for its simplicity of construction [6]. Also in the 1980s, Hindmarsh and Rose produced a system of three differential equations to model neuronal activity [7]. Lastly, in the 1990s, René Thomas developed a system of three differential equations that could be used to model a particle's motion in a 3D lattice of forces [8].

Early examples of the application of chaotic systems for music date back to the late 1990s with Di Scipio's work on iterated nonlinear functions [9,10]. There, Di Scipio investigates difference equations based on the sine map model for the generation of spectrally rich synthetic sounds and dynamical behaviours and simulations of auditory environmental events. In the 2000s, we have investigations by Bilotta et al., where the Chua oscillator is explored extensively for musical and artistic applications [11, 12]. More recent works on musical applications of chaotic systems can be found in [13] and [14]. Mudd, in particular, develops a form of modified Duffing equation where the dynamics of the system is guided through a band-pass filter bank; the equation and the filters are coupled through an added outer feedback loop. Finally, a survey of feedback-based music describing several approaches that can all be grouped within the realm of nonlinear iterated functions is available in [15].

Since Bilotta et al. and Mudd have carried out research on Chua's circuit and Duffing equation, in the next section, we will focus on the remaining differential equations mentioned earlier: Lotka-Volterra, van der Pol, Lorenz, Rössler, Hindmarsh-Rose, and Thomas.

2. COMPLEX SOUND GENERATORS

This section will discuss the implementation of complex sound generators based on modified chaotic differential equations, and we will present results to show their musical potential.

Considering first-order differential equations, we can obtain a generalisation of the modified systems discussed here. Let y be a vector of functions, let x be an input vector, let F be a vector of functions of y and x; let C be a vector of constraining functions. Written in differential form with respect to time, we have that:

$$\frac{\partial y(t)}{\partial t} = C(F(x(t), y(t))) \tag{1}$$

where

$$C(z) = B(l \cdot S(z/l)) \tag{2}$$

with B and S being, respectively, vectors of first-order DC-blockers and saturating nonlinearities with arbitrary saturation threshold piloted by the parameter l. The saturation threshold becomes a key parameter for the interaction with the oscillators, while the overall output can be normalised to unity peak amplitudes for digital audio by merely dividing by l. While several types of bounded saturators are available [16], here, we will focus on the well-known hyperbolic tangent function. Note that the input vector can be used to set the system's initial conditions or as continuous perturbation through signals. In fact, these systems can also be deployed as nonlinear distortion units when operating under non-self-oscillating conditions.

The working concept for the constraining function is inherent to the nature of the differential equations studied here. Firstly, tendencies towards fixed-point attractors [17] will be counteracted by the DC-blockers, which will result in self-oscillating behaviours due to mutually compensating mechanisms. Secondly, tendencies towards unbounded exponential growth will be turned into fixed-points by the saturating functions, which will subsequently be contrasted by the DC-blocker, favouring self-oscillation and evolutions at timbral and formal time scales. The resulting modified systems are structurally stable for any parameters values and can be explored through a larger state variable space. Visual phase space analysis [18, 19] shows novel and enhanced complex behaviours. Note that while the DC-blockers have a fixed cut-off of ten Hz for the examples below, their frequency, when using sub-audio cut-off values, can be a key parameter for formal-level variations. In particular, the parameter sets the responsiveness for the fixed-point counterbalancing mechanism.

The differential equations and their respective discretised models will be shown later. The implementation of the generators in the Faust[1] language is available on `http://github.com/dariosanfilippo`. Audio examples are available on `http://soundcloud.com/dario-sanfilippo`.

2.1 Remarks on nonlinearities and aliasing distortion

Aliasing distortion is a common problem in digital audio, and it can significantly compromise the quality of computer-generated sound and music. For example, several techniques have been developed to overcome issues related to aliasing in digital oscillators of classic analogue waveforms [20–22]. Processing audio signals through nonlinearities, too, can result in high aliasing distortion depending on the nature of the process. Particularly for saturators such as the hyperbolic tangent function, we have the generation of odd harmonics given by the odd symmetry of the function, while their strength is proportional to the amplitude of the input signal. A standard technique to counterbalance aliasing distortion, in general, is oversampling, while more efficient techniques based on antiderivatives have been explicitly developed for nonlinear processing [23–25].

Unlike aliasing distortion in digital oscillators or saturators used in feedforward configurations, aliasing distortion in recursive systems such as those discussed here acquires a systemic role within global behaviours and has proven to be worthy of exploration for musical purposes. Nonetheless, applications of antialiasing techniques will be investigated in the future to realise more accurate approximations of the models in this paper. The author already operates these systems at 192 kHz sample-rate to reduce distortion – a low-order oversampling ratio achievable on most hardware devices when oversampling is not available via software.

2.2 Discrete models

For the discretisation of continuous-time systems, we will rely on the Euler method for first-order approximation [26]. For simplicity, the discrete equations will only include approximations of the continuous differential equations and omit the input signals and the constraining functions showed in (1) and implemented in the Faust code. Note that to clarify the discretisation, the dimensions of the systems were named after the original differential equations rather than following the general formula (1). The numerical simulations below are in double-precision; the parameters were chosen according to trial-and-error processes.

2.2.1 Lotka-Volterra

The Lotka-Volterra system consists of two differential equations with four parameters. The equations model biological systems and interaction dynamics between predator-prey couples. The four parameters, assumed to be positive real values, govern critical aspects of a biological system such as the population growth of preys and predators and the degree of interaction between the two:

$$\begin{aligned} \frac{\partial x}{\partial t} &= \alpha x - \beta xy \\ \frac{\partial y}{\partial t} &= \delta xy - \gamma y \end{aligned} \quad . \tag{3}$$

The discrete model approximation is given by:

$$\begin{aligned} x[n] &= x[n-1] + \partial t(\alpha x[n-1] - \beta x[n-1]y[n-1]) \\ y[n] &= y[n-1] + \partial t(\delta x[n-1]y[n-1] - \gamma y[n-1]) \end{aligned} \tag{4}$$

where ∂t is the integration step.

Stability analysis and characteristics of Lotka-Volterra systems have been investigated largely. A notable publication describing global behaviours can be found in [27]. Be-

[1] `http://faust.grame.fr`

low, we can see the output of the modified Lotka-Volterra equations using parameters in unstable regions. Specifically, the parameters for this example are: $\alpha = 4$, $\beta = 1$, $\delta = 2$, and $\gamma = 1$, with the initial conditions set to 1 for both equations. The saturation threshold is 30, and the integration step, ∂t, is .1. In the non-modified equations, these parameters would result in an unbounded exponential growth that would exceed representability in floating-point double-precision after a few iterations.

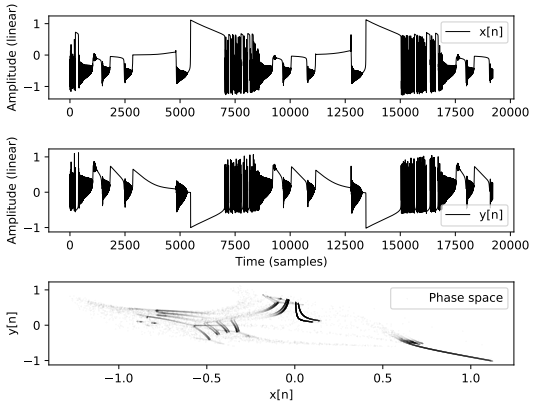


Figure 1. Modified Lotka-Volterra system outputs and phase space with $x_0 = 1$, $y_0 = 1$, $\alpha = 4$, $\beta = 1$, $\delta = 2$, $\gamma = 1$, $l = 30$, and $\partial t = .1$.

2.2.2 *Van der Pol*

The van der Pol system is a second-order differential equation describing a non-conservative oscillator with single nonlinear damping parameter. Dutch engineer Balthasar van der Pol discovered the system in the 1920s and found out that, under specific conditions, the oscillator would exhibit chaotic behaviours. The differential equation for the oscillator is:

$$\frac{\partial^2 x}{\partial t^2} - \mu(1 - x^2)\frac{\partial x}{\partial t} + x = 0 \quad , \tag{5}$$

where $\mu > 0$ represents the strength of the nonlinearity. Using Liénard's transformation [28]

$$y = x - \frac{x^3}{3} - \frac{\partial x}{\partial t}\frac{1}{\mu} \quad , \tag{6}$$

we can rewrite the Van der Pol second-order equation as a two-dimensional, first-order system as follows:

$$\begin{aligned} \frac{\partial x}{\partial t} &= \mu\left(x - \frac{x^3}{3} - y\right) \\ \frac{\partial y}{\partial t} &= \frac{x}{\mu} \end{aligned} \quad . \tag{7}$$

The discrete model for the Van der Pol oscillator is then:

$$\begin{aligned} x[n] &= x[n-1] + \\ &\partial t\left(\mu\left(x[n-1] - \frac{x^3[n-1]}{3} - y[n-1]\right)\right) \\ y[n] &= y[n-1] + \partial t\left(\frac{x[n-1]}{\mu}\right) \end{aligned} \quad . \tag{8}$$

In figure 2, we can see the response the system with constant input signals equal to 1, $l = 1.5733$, $\partial t = .904001$, and $\mu = .664$.

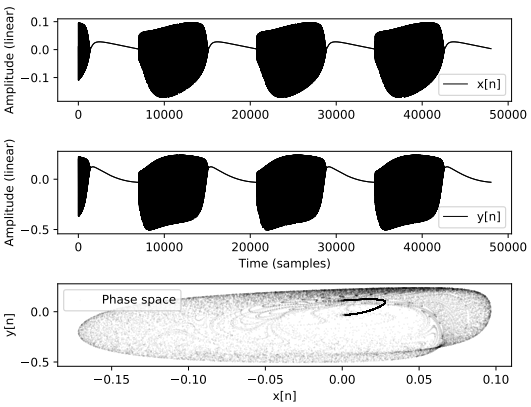


Figure 2. Modified Van der Pol system outputs and phase space with constant input signals equal to 1, $l = 1.5733$, $\partial t = .904001$, and $\mu = .664$.

2.2.3 *Lorenz*

The Lorenz system is one of the most influential models for deterministic chaos. It was initially developed in the 1960s by Edward Lorenz as a simplified model for atmospheric convection. The model is a first-order system of three differential equations relating convection to horizontal and vertical temperature variations [29]. The parameters σ, ρ, and β govern the relationships between the quantities. Lorenz initially chose the values $\sigma = 10$, $\rho = 8/3$, and $\beta = 28$, a parameters region displaying chaotic behaviours and producing the popular butterfly-like phase space. The system is given by:

$$\begin{aligned} \frac{\partial x}{\partial t} &= \sigma(y - x) \\ \frac{\partial y}{\partial t} &= x(\rho - z) - y \\ \frac{\partial z}{\partial t} &= xy - \beta z \end{aligned} \quad . \tag{9}$$

The discrete model is given by:

$$
\begin{aligned}
x[n] &= x[n-1] + \partial t(\sigma(y[n-1] - x[n-1])) \\
y[n] &= y[n-1] + \\
&\quad \partial t(x[n-1](\rho - z[n-1]) - y[n-1]) \\
z[n] &= z[n-1] + \partial t(x[n-1]y[n-1] - \beta z[n-1])
\end{aligned}
\tag{10}
$$

Dynamical behaviours of the Lorenz system are often investigated by keeping σ and ρ fixed to the values originally proposed by Lorenz while varying β in a range between 0 and 30 [30]. In the example showed in figures 3 and 4, we can see chaotic behaviours with $\sigma = 10$, $\rho = 2.67$, and $\beta = -10$. The integration step, ∂t, is set to 0.022001, and the saturation limit is $l = 143.810806$.

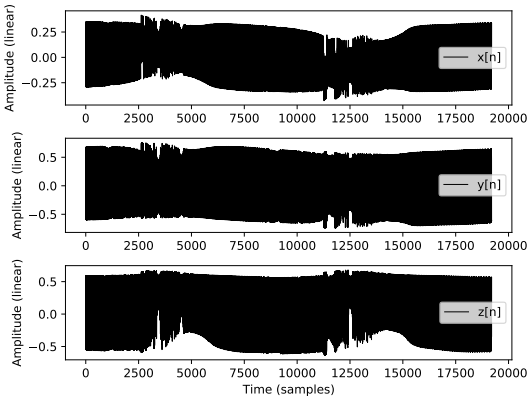


Figure 3. Modified Lorenz system impulse response outputs with $\sigma = 10$, $\rho = 2.67$, and $\beta = -10$. The integration step, ∂t, is set to 0.022001, and the saturation limit is $l = 143.810806$.

2.2.4 Rössler

The Rössler system was developed in the 1970s as a simplified model for continuous chaos with similar properties to those of the Lorenz system. Based on the Poincaré-Bendixson theorem for which a three-dimensional manifold is a minimum requirement for chaos, Rössler developed a first-order system of three differential equations with three parameters and minimal nonlinearity inspired by the geometry of relaxation-type systems [31]:

$$
\begin{aligned}
\frac{\partial x}{\partial t} &= -y - z \\
\frac{\partial y}{\partial t} &= x + ay \\
\frac{\partial z}{\partial t} &= bx - cz + xz
\end{aligned}
\tag{11}
$$

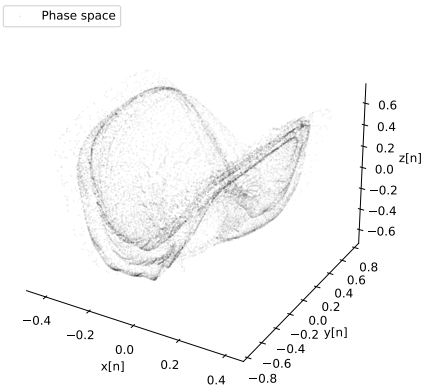


Figure 4. Modified Lorenz system impulse response phase space with $\sigma = 10$, $\rho = 2.67$, and $\beta = -10$. The integration step, ∂t, is set to 0.022001, and the saturation limit is $l = 143.810806$.

An example of chaotic behaviour is with the parameters $a = 0.38$, $b = 0.3$, and $c = 4.820$. The discrete model is given by:

$$
\begin{aligned}
x[n] &= x[n-1] + \partial t(-y[n-1] - z[n-1]) \\
y[n] &= y[n-1] + \partial t(x[n-1] + ay[n-1]) \\
z[n] &= z[n-1] + \\
&\quad \partial t(bx[n-1] - cz[n-1] + x[n-1]z[n-1])
\end{aligned}
\tag{12}
$$

In figures 5 and 6, we can see chaotic behaviours of the modified system through its outputs and phase space for $a = 0.776$, $b = 2.524$, $c = 13.98$, $\partial t = 2.075001$, and $l = 21.2554$.

2.2.5 Hindmarsh-Rose

The Hindmarsh-Rose system was developed in the 1980s as a model describing neuronal activity to study the behaviours of the membrane potential. Real neurons exhibit various behaviours ranging from quiescence to regular and irregular spiking-bursting outputs, and analysis of the model has shown that it can reproduce such behaviours correctly [32]. The model consists of three first-order differential equations and has eight parameters in total:

$$
\begin{aligned}
\frac{\partial x}{\partial t} &= y + \phi(x) - z + I \\
\frac{\partial y}{\partial t} &= \psi(x) - y \\
\frac{\partial z}{\partial t} &= r(s(x - x_R) - z)
\end{aligned}
\tag{13}
$$

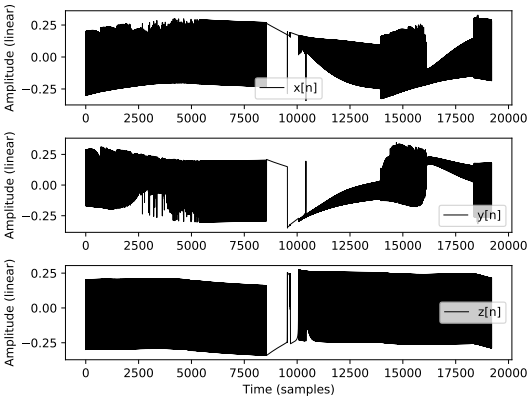


Figure 5. Modified Rössler system outputs for $a = 0.776$, $b = 2.524$, $c = 13.98$, $\partial t = 2.075001$, and $l = 21.2554$.

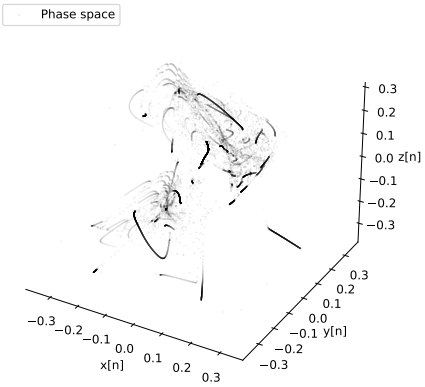


Figure 6. Modified Rössler system phase space for $a = 0.776$, $b = 2.524$, $c = 13.98$, $\partial t = 2.075001$, and $l = 21.2554$.

where

$$\begin{aligned}\phi(x) &= -ax^3 + bx^2 \\ \psi(x) &= c - dx^2\end{aligned} \quad . \tag{14}$$

The discrete equations are:

$$\begin{aligned}x[n] &= x[n-1] + \\ &\quad \partial t(y[n-1] + \phi(x[n-1]) - z[n-1] + I) \\ y[n] &= y[n-1] + \partial t(\psi(x[n-1]) - y[n-1]) \\ z[n] &= z[n-1] + \partial t(r(s(x[n-1] - x_R) - z[n-1]))\end{aligned} \tag{15}$$

The neuronal model is often analysed by keeping some of the parameters fixed while varying the remaining ones. In [33], we have $a = 1$, $b = 3$, $c = -3$, $d = 5$, $s = 4$, and $I = 5$. In [34], instead, we have $a = 1$, $b = 3$, $c = 1$, $d = 5$, $s = 4$, and $x_R = 8/5$, where chaotic regions are characterised in terms of Lyapunov analysis with positive peaks in the Lyapunov exponent being detected for $r = .0021$ and $I \approx 3.295$. In the examples of the modified Hindmarsh-Rose system, we will choose different parameters to show chaotic behaviours in alternative regions. In figures 7 and 8, we can see the outputs and phase space of the modified Hindmarsh-Rose model with $a = 1$, $b = -5.864$, $c = -20$, $d = -5.656$, $r = -.192$, $s = 3.104$, $I = -6.836$, and $x_R = 8.792$, with $l = 5.3989$ and $\partial t = .299001$.

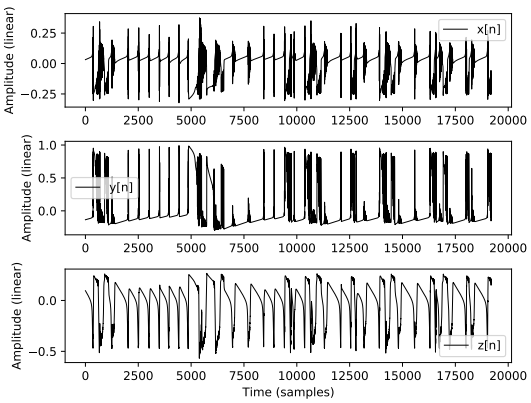


Figure 7. Modified Hindmarsh-Rose system phase space with $a = 1$, $b = -5.864$, $c = -20$, $d = -5.656$, $r = -.192$, $s = 3.104$, $I = -6.836$, and $x_R = 8.792$, with $l = 5.3989$ and $\partial t = .299001$.

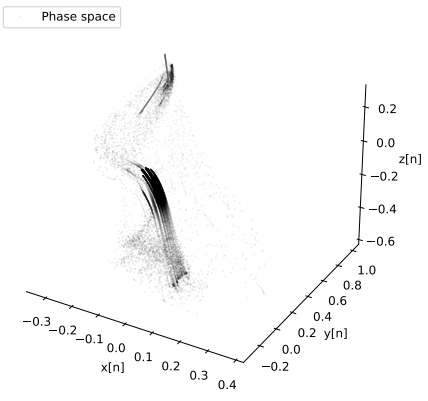


Figure 8. Modified Hindmarsh-Rose system phase space with $a = 1$, $b = -5.864$, $c = -20$, $d = -5.656$, $r = -.192$, $s = 3.104$, $I = -6.836$, and $x_R = 8.792$, with $l = 5.3989$ and $\partial t = .299001$.

2.2.6 *Thomas*

The Thomas system was developed in the late 1990s by René Thomas. It is a relatively simple three-dimensional first-order differential equation with one parameter representing damped particles moving in a 3D lattive of forces [35]. The system is defined by the following equations:

$$\begin{aligned} \frac{\partial x}{\partial t} &= \sin(y) - bx \\ \frac{\partial y}{\partial t} &= \sin(z) - by \\ \frac{\partial z}{\partial t} &= \sin(x) - bz \end{aligned} \quad . \tag{16}$$

The discrete model is given by:

$$\begin{aligned} x[n] &= x[n-1] + \partial t(\sin(y[n-1]) - bx[n-1]) \\ y[n] &= y[n-1] + \partial t(\sin(z[n-1]) - by[n-1]) \\ z[n] &= z[n-1] + \partial t(\sin(x[n-1]) - bz[n-1]) \end{aligned} \quad . \tag{17}$$

In figures 9 and 10, we can see low-level and high-level complex oscillations with $b = .008$, $l = 5.3989$, and $\partial t = .841001$.

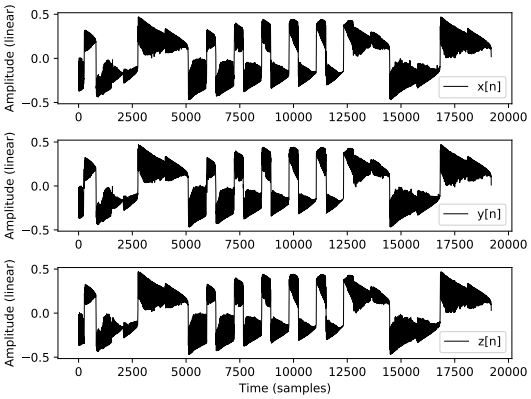


Figure 9. Modified Thomas system outputs with $b = .008$, $l = 5.3989$, and $\partial t = .841001$.

3. CONCLUSION

In this paper, we have discussed the investigation of modified differential equations to generate complex audio signals for musical applications. While the original differential equations studied here provided complex behaviours for specific configurations of the parameters, those systems only allowed for a limited state variable exploration due to possible stability issues. The modified equations proposed here deploy a constraining mechanism based on cascaded

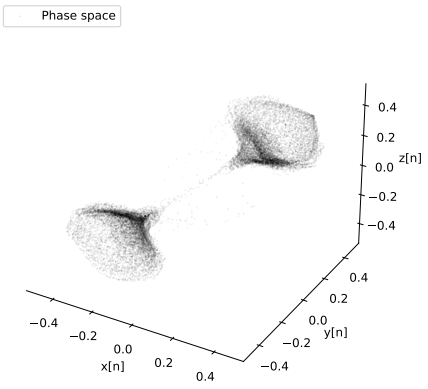


Figure 10. Modified Thomas system phase space with $b = .008$, $l = 5.3989$, and $\partial t = .841001$.

saturating nonlinearities and DC-blocking units that make the systems structurally stable and produce output signals in ranges suitable for digital audio applications. Most importantly, the constraining mechanism provides a low-level configuration establishing an interplay between positive and negative feedback relations, which results in nontrivial dynamical behaviours at timbral and formal time scales. The modified systems can be explored through a larger state variable for novel and enhanced complex sound generation and can be deployed in computer music environments in general and, more specifically, human-machine interaction performances requiring agency and autonomy as discussed in [36, 37].

Future works include applying higher-order methods for the solution of differential equations [38] and antialiased nonlinearities as discussed in 2.1 for a better approximation of the models, and advanced adaptation techniques for time-variant generators with increased complexity following [39–41]. Taking advantage of the implementation that allows for input signals to be used as external perturbation, the individual systems can be combined into networks of interacting components to explore emergent behaviours resulting from their synergy.

Finally, an extension of this work will be an analytical examination and comparison between the classic chaotic differential equations and the proposed modified systems. By means of Lyapunov exponent analysis [42], it will be possible to determine the effects of the constraining infrastructures concerning the sensitivity to initial conditions of the systems, while phase space analysis using musically-relevant dimensions such loudness, spectral centroid, and noisiness can provide a tool for the comparison of higher-level complexity in the original and modified systems.

4. REFERENCES

[1] A. J. Lotka, "Contribution to the theory of periodic reactions," *The Journal of Physical Chemistry*, vol. 14, no. 3, pp. 271–274, 2002.

[2] G. Duffing, *Erzwungene Schwingungen bei veränder-*

licher Eigenfrequenz und ihre technische Bedeutung. Vieweg, 1918, no. 41-42.

[3] B. Van der Pol, "Lxxxviii. on "relaxation-oscillations"," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 978–992, 1926.

[4] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.

[5] O. E. Rössler, "An equation for continuous chaos," *Physics Letters A*, vol. 57, no. 5, pp. 397–398, 1976.

[6] T. Matsumoto, "A chaotic attractor from chua's circuit," *IEEE Transactions on Circuits and Systems*, vol. 31, no. 12, pp. 1055–1058, 1984.

[7] J. L. Hindmarsh and R. Rose, "A model of neuronal bursting using three coupled first order differential equations," *Proceedings of the Royal society of London. Series B. Biological sciences*, vol. 221, no. 1222, pp. 87–102, 1984.

[8] R. Thomas, "Deterministic chaos seen in terms of feedback circuits: Analysis, synthesis," labyrinth chaos"," *International Journal of Bifurcation and Chaos*, vol. 9, no. 10, pp. 1889–1905, 1999.

[9] A. Di Scipio and I. Prignano, "Synthesis by functional iterations. a revitalization of nonstandard synthesis," *Journal of New Music Research*, vol. 25, no. 1, pp. 31–46, 1996.

[10] A. Di Scipio, "Synthesis of environmental sound textures by iterated nonlinear functions," in *Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99)*, 1999, pp. 109–117.

[11] E. Bilotta and P. Pantano, "The language of chaos," *International Journal of Bifurcation and Chaos*, vol. 16, no. 03, pp. 523–557, 2006.

[12] ——, *Gallery Of Chua Attractors, A (With Dvd-rom)*. World Scientific, 2008, vol. 61.

[13] T. Clutterbuck, T. Mudd, and D. Sanfilippo, "A practical and theoretical introduction to chaotic musical systems," in *In Proc. of the International Conference on Live Interfaces, eds. Thor Magnusson, Chris Kiefer, and Sam Duffy.*, 2016.

[14] T. Mudd, "Between chaotic synthesis and physical modelling: Instrumentalising with gutter synthesis," in *Procedings of the Conference on Compuation, Communication, Aesthetics and X (xCoAx)*, 2019, pp. 217–228.

[15] D. Sanfilippo and A. Valle, "Feedback systems: An analytical framework," *Computer Music Journal*, vol. 37, no. 2, pp. 12–27, 2013.

[16] V. Zavalishin, "The art of va filter design," *Native Instruments, Berlin, Germany*, 2012.

[17] J. Gleick, *Chaos: Making a new science*. Open Road Media, 2011.

[18] D. D. Nolte, "The tangled tale of phase space," *Physics today*, vol. 63, no. 4, pp. 33–38, 2010.

[19] H. Sayama, *Introduction to the modeling and analysis of complex systems*. Open SUNY Textbooks, 2015.

[20] T. Stilson and J. O. Smith III, "Alias-free digital synthesis of classic analog waveforms," in *ICMC*, 1996.

[21] J. Nam, V. Valimaki, J. S. Abel, and J. O. Smith, "Efficient antialiasing oscillator algorithms using low-order fractional delay filters," *IEEE transactions on audio, speech, and language processing*, vol. 18, no. 4, pp. 773–785, 2009.

[22] V. Valimaki, J. Nam, J. O. Smith, and J. S. Abel, "Alias-suppressed oscillators based on differentiated polynomial waveforms," *IEEE Transactions on audio, speech, and language processing*, vol. 18, no. 4, pp. 786–798, 2009.

[23] J. D. Parker, V. Zavalishin, and E. Le Bivic, "Reducing the aliasing of nonlinear waveshaping using continuous-time convolution," in *Proc. Int. Conf. Digital Audio Effects (DAFx-16), Brno, Czech Republic*, 2016, pp. 137–144.

[24] S. Bilbao, F. Esqueda, J. D. Parker, and V. Välimäki, "Antiderivative antialiasing for memoryless nonlinearities," *IEEE Signal Processing Letters*, vol. 24, no. 7, pp. 1049–1053, 2017.

[25] M. Holters, "Antiderivative antialiasing for stateful systems," *Applied Sciences*, vol. 10, no. 1, p. 20, 2020.

[26] S. Bilbao, *Numerical sound synthesis: finite difference schemes and simulation in musical acoustics*. John Wiley & Sons, 2009.

[27] Y. Takeuchi, *Global dynamical properties of Lotka-Volterra systems*. World Scientific, 1996.

[28] D. Kaplan and L. Glass, *Understanding nonlinear dynamics*. Springer Science & Business Media, 1997.

[29] C. Sparrow, *The Lorenz equations: bifurcations, chaos, and strange attractors*. Springer Science & Business Media, 2012, vol. 41.

[30] W. Song and J. Liang, "Difference equation of lorenz system," *International Journal of Pure and Applied Mathematics*, vol. 83, no. 1, pp. 101–110, 2013.

[31] P. Gaspard, "Rössler systems," *Encyclopedia of nonlinear science*, vol. 231, pp. 808–811, 2005.

[32] M. Storace, D. Linaro, and E. de Lange, "The hindmarsh–rose neuron model: bifurcation analysis and piecewise-linear approximations," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 18, no. 3, p. 033128, 2008.

[33] A. Shilnikov and M. Kolomiets, "Methods of the qualitative theory for the hindmarsh–rose model: A case study–a tutorial," *International Journal of Bifurcation and chaos*, vol. 18, no. 08, pp. 2141–2168, 2008.

[34] X.-J. Wang, "Genesis of bursting oscillations in the hindmarsh-rose model and homoclinicity to a chaotic saddle," *Physica D: Nonlinear Phenomena*, vol. 62, no. 1-4, pp. 263–274, 1993.

[35] J. C. Sprott and K. E. Chlouverakis, "Labyrinth chaos," *International Journal of Bifurcation and Chaos*, vol. 17, no. 06, pp. 2097–2108, 2007.

[36] D. Sanfilippo and A. Di Scipio, "Environment-mediated coupling of autonomous sound-generating systems in live performance: An overview of the machine milieu project," in *Proceedings of the 14th Sound and Music Computing Conference, Espoo, Finland*, 2017, pp. 5–8.

[37] A. Di Scipio and D. Sanfilippo, "Defining ecosystemic agency in live performance," *Array*, pp. 28–43, 2019.

[38] D. Medine, "Dynamical systems for audio synthesis: Embracing nonlinearities and delay-free loops," *Applied Sciences*, vol. 6, no. 5, p. 134, 2016.

[39] D. Sanfilippo, "Time-variant infrastructures and dynamical adaptivity for higher degrees of complexity in autonomous music feedback systems: the Order from Noise (2017) project," *Musica/Tecnologia*, vol. 12, no. 1, pp. 119–129, 2018.

[40] ——, "Time-domain algorithms for low-level and high-level information processing," *Computer Music Journal (pending peer-review)*, 2021.

[41] ——, "Complex adaptation in audio feedback networks for the synthesis of music and sounds," *Computer Music Journal (pending peer-review)*, 2021.

[42] J. B. Dingwell, "Lyapunov exponents," *Wiley encyclopedia of biomedical engineering*, 2006.

REAL-TIME TIMBRE TRANSFER AND SOUND SYNTHESIS USING DDSP

Francesco GANIS (fganis20@student.aau.dk) [1], **Erik F. KNUDSEN** (eknuds20@student.aau.dk) [1], **Søren V. K. LYSTER** (slyste20@student.aau.dk) [1], **Robin OTTERBEIN** (rotter20@student.aau.dk) [1], **David SÜDHOLT** (dsudho20@student.aau.dk) [1], and **Cumhur ERKUT** (cer@create.aau.dk) (0000-0003-0750-1919) [1]

[1] *Sound and Music Computing, Department of Architecture, Design, and Media Technology*, **Aalborg University**, A.C. Meyers Vænge, Copenhagen, DK-2450 Denmark

ABSTRACT

Neural audio synthesis is an actively researched topic, having yielded a wide range of techniques that leverages machine learning architectures. Google Magenta elaborated a novel approach called Differential Digital Signal Processing (DDSP) that incorporates deep neural networks with preconditioned digital signal processing techniques, reaching state-of-the-art results especially in timbre transfer applications. However, most of these techniques, including the DDSP, are generally not applicable in real-time constraints, making them ineligible in a musical workflow. In this paper, we present a real-time implementation of the DDSP library embedded in a virtual synthesizer as a plug-in that can be used in a Digital Audio Workstation. We focused on timbre transfer from learned representations of real instruments to arbitrary sound inputs as well as controlling these models by MIDI. Furthermore, we developed a GUI for intuitive high-level controls which can be used for post-processing and manipulating the parameters estimated by the neural network. We have conducted a user experience test with seven participants online. The results indicated that our users found the interface appealing, easy to understand, and worth exploring further. At the same time, we have identified issues in the timbre transfer quality, in some components we did not implement, and in installation and distribution of our plugin. The next iteration of our design will address these issues.

1. INTRODUCTION

Sound synthesizers have been widely used in music production since the late 50s. Because of their inner complexity, many musicians and producers polish presets' parameters until they reach the desired sound. This procedure is time-consuming and sometimes results in failed attempts to achieve a desired sound.

Much research has been done in the area of automating the generation of these sounds through the aid of machine learning and neural networks. Common approaches included directly generating the waveform in the time domain [1] or predicting synthesis parameters based on hand-picked analysis features [2]. In their 2020 paper on Differentiable Digital Signal Processing (DDSP) [3], Engel et al. proposed a novel approach to neural audio synthesis. Rather than generating signals directly in the time or frequency domain, DDSP offers a complete end-to-end toolbox consisting of a synthesizer based on Spectral Modeling Synthesis (SMS) [4], and an autoencoder neural network architecture that takes care of both extracting analysis features and predicting synthesis parameters.

The authors of the DDSP paper released a public demonstration of "tone transfer" [1], allowing the user to upload their own recordings, select from a list of models trained on various instruments and "transfer" their recorded melodies to the sound of a trumpet, a violin etc. Based on these, we implemented the DDSP back-end as a virtual instrument playable in real-time. Figure 1 shows the GUI of our synthesizer.

This paper documents the background, our requirement-driven design and implementation approach, including model components and training, the GUI design, and user experience evaluation. The structure of this paper follows these main topics in order.

Besides our contribution to the real-time neural audio synthesis and its user experience evaluation, we release our real-time MATLAB and JUCE implementations at `https://github.com/SMC704/juce-ddsp` and `https://github.com/SMC704/matlab-ddsp`, respectively. We also provide a demonstration video at `https://share.descript.com/view/hXAZLCPJNqm`.

Figure 1. Our real-time DDSP Synthesizer GUI.

[1] `https://sites.research.google/tonetransfer`, last accessed on 2020-11-30

2. BACKGROUND

In addition to the DDSP paper [3], our work is inspired by the commercially produced additive synthesizer called *Razor* by Native Instruments [5]. Razor's core consists of a powerful additive synthesizer and features various modulation options for manipulating the sound output. What is especially interesting about Razor is that every modulation option (e.g. filters, stereo imaging, reverbs and delays) is actually modulating individual partial harmonics (non-integer multiples of the fundamental frequency) in the additive synthesis engine. Furthermore, Razor enables musicians and producers to intuitively control partials via different parameters while relying on a visual representation of partial manipulation. We therefore focused on the harmonic and the stochastic components of the DDSP.

2.1 Harmonic Oscillator / Additive Synthesizer

The additive synthesizer is the main core of the whole synthesis and is responsible for generating all the harmonic components of the reconstructed sound. The output is characterized by the sum of several harmonic integer multiples of the fundamental frequency f_0:

$$f_k(n) = k \cdot f_0(n). \tag{1}$$

In order to generate the harmonics, we can implement k oscillators in the discrete time:

$$x(n) = \sum_{k=1}^{K} A_k(n) \cdot \sin(\phi_k(n)), \tag{2}$$

where $A_k(n)$ is the time-varying amplitude of the k_{th} sinusoidal component and $\phi_k(n)$ is its instantaneous phase. $\phi_k(n)$ is obtained by integrating the instantaneous frequency $f_k(n)$ [3]:

$$\phi_k(n) = 2\pi \sum_{m=0}^{n} f_k(m) + \phi_{0,k}. \tag{3}$$

The only two parameters necessary to control the synthesizer are the frequency $f_0(n)$ and the harmonic amplitudes $A_k(n)$. These are retrieved directly from the input sound using the encoder contained in the autoencoder network. As reported in [3], the network outputs are scaled and normalized to fall within an interpretable value range for the synthesizer.

2.2 Filtered Noise, Subtractive Synthesizer, and Reverb

The subtractive synthesis is used to recreate the non-harmonic part of natural sounds. The parameters necessary to obtain a frequency-domain transfer function of a linear time-variant finite impulse response (LTV-FIR) filter are retrieved from the neural network in frames that are subsets of the input signal. The corresponding impulse responses (IRs) are calculated and a windowing function is applied. The windowed IRs are then convolved with white noise via transformation to and multiplication in the frequency domain. Another LTV-FIR filter acts as a reverberator, performing esentially a convolution reverb in the frequency domain.

2.3 Research question & design requirements

Based on this background we have formulated the following research question: How can we develop a playable software instrument, based on the DDSP library, that would: a) allow customization of model-estimated synth parameters through top-level macro controls, b) enable existing workflow-integration in Digital Audio Workstations (DAWs), and c) facilitate a simple approach for beginners without limiting usability for expert music producers?

Based on this research question, we have identified five user-objectives [6], matched them with a solution, and reformulated them as design requirements that address the following functionality: building a playable real-time software-instrument plugin that supports different composition techniques by having audio and MIDI input modes. The instrument must include at least four models which serve the purpose of estimating synthesizer parameters to output a desired sound. Finally, the instrument must include graphical user interface components providing intuitive controls for the manipulation of synthesizer and effect parameters. The design requirements are documented on Table 1.

3. DESIGN & IMPLEMENTATION

3.1 Architecture overview

To meet our criteria of creating a real-time software instrument, we decided to build the plugin in C++ using the JUCE application framework [2] . With JUCE, we had a multi-platform supported audio plugin template that was handling MIDI and audio inputs and outputs. This allowed us to mainly focus on the audio processing and GUI.

Creating a real-time implementation of the non-real-time DDSP library posed some immediate challenges. To analyze and understand these challenges we decided to start by doing a direct translation of the additive and subtractive synthesizers from the DDSP library into MATLAB. The synthesizers could then be changed into real-time implementations and tested. In order to use our MATLAB implementation in the JUCE framework, we used inbuilt MATLAB tools to generate C++ code.

We transformed the autoencoder models pretrained by Google into models that could be used to estimate synthesizer parameters directly from our plugin's user input.

A general overview of this architecture can be seen in figure 2. The following sections will discuss each component in more detail.

3.1.1 Synth in MATLAB

MATLAB's environment and visualization tools gave us access to quick prototyping and testing. This allowed us to do the implementation over multiple iterations. We tested our synthesizers' compatibility with the predicted parameters from the DDSP models by invoking the encoders and decoders in isolation through MATLAB's Python interface.

At first we implemented the non-real-time synthesis algorithms of the DDSP library. Then the synthesizers were changed to real-time, i.e., synthesizing a single frame at

[2] `https://juce.com/`, last accessed on 2020-12-15

#	User Obj.	Solution	Design Requirement
1	Provide a new playable instrument for unique sound generation and inspiration	Real-time implementation	*Must work in real-time as a playable software instrument.*
2	Conveniently integrate into existing workflows	Plugin format application	*Must be implemented as a software plugin.*
3	Adapt to different composition methods	Allow line and MIDI input	*Must allow switching between Line and MIDI input.*
4	Easy fast unique sound generation	Choose models for sound generation	*Must implement at least four pre-trained models.*
5	Convenient customizability of sounds	Tweakable parameters that effects the audio output	*Must include GUI components for intuitive manipulation of synth and effects parameters.*

Table 1. Documentation of Design Requirements

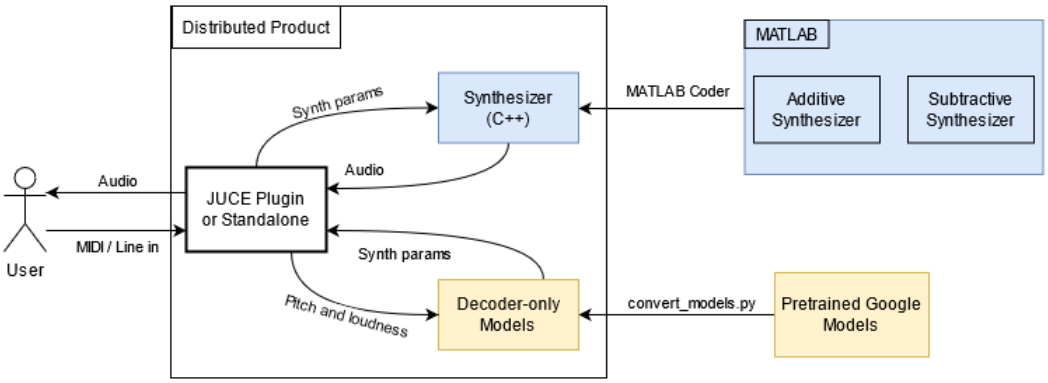


Figure 2. Schematic overview of the project architecture.

a time. Using the MATLAB Audio Test Bench, we could then test the functionality of the synthesizer components and parameters with real-time audio and varying sample rate and buffer size. The last iterations consisted of optimizing the code with the constraints of real-time audio processing on CPUs.

3.1.2 MATLAB to C++

Using the MATLAB coder tool [3] we were able to generate C++ functions from the MATLAB code. For the simplest integration between the generated C++ functions and the JUCE plugin we chose to limit the function inputs and outputs to built-in and derived C++ data types. This required our MATLAB functions to have fixed-sized inputs and outputs. We decided on a maximum input/output size of 4096 double-precision floating point numbers, this being the maximum buffer size the plugin could handle.

A helper file was created to ensure code consistency, allowing the user and MATLAB coder to verify the functions with different inputs. Having this setup made it easy to go back to the MATLAB code and generate updated C++ functions without breaking the JUCE plugin.

3.1.3 TensorFlow in C++

Running the DDSP TensorFlow implementation in a real-time audio application is a heavy computational challenge. Moving from TensorFlow in Python to the TensorFlow C API [4] allowed us to integrate the models into the C++ codebase. By moving the TensorFlow computations to a separate thread, we load the models, set the inputs, run the parameter estimation and save the outputs, without experiencing buffer underruns in the main audio processing thread.

3.1.4 Input signals

The DDSP autoencoder needs the input values *fundamental frequency* (f_0) and *loudness*. Since we allow both MIDI and line-in audio, two separate implementations are needed to calculate these values, which were first created in MATLAB. In the C++ implementation we chose the YIN pitch tracking algorithm [7] from the C library Aubio [8], since it yielded more precise results.

3.2 Training models

DDSP autoencoders are trained to reconstruct waveforms with minimal perceptual loss. Similarity of the raw waveform however is not a good indicator for perceptual similarity, which is why the DDSP library makes use of multi-scale spectral loss [3]. The total reconstruction loss is the sum of multiple spectral losses, i.e., the difference of the magnitude spectrograms, over various time scales. Moreover, the linear magnitude losses are sensitive to the peaks, whereas logarithmic magnitude losses are sensitive to the quiet regions of the signals. Therefore, the sum of losses $L = \sum_i L_i$ in DDSP are calucated in six different frame sizes by

$$L_i = ||S_i - \hat{S}_i||_1 + ||log(S_i) - log(\hat{S}_i)||_1. \quad (4)$$

[3] https://se.mathworks.com/products/matlab-coder.html, last accessed on 2020-12-15

[4] https://www.tensorflow.org/install/lang_c, last accessed on 2020-12-15

3.2.1 Pre-trained models

Next to the *tone transfer* website mentioned in the introduction, the authors of the DDSP paper also published a Jupyter Notebook Demo on Google Colab called *timbre transfer.* [5] We accessed the available checkpoint files for violin, flute, tenor saxophone and trumpet from this notebook for our real-time implementation of the timbre transfer. However, we were not immediately able to use them in the JUCE plugin. The DDSP models are trained using TensorFlow's *eager execution mode*, while the TensorFlow C API is constructed around *graph mode*. Additionally, since we required the models to be controllable by MIDI input, we needed direct access to the decoder part of the model instead of supplying audio to the encoder.

The `convert_models.py` script from the Python folder of the plugin code repository deals with these requirements by loading the eager model from the downloaded checkpoint file, constructing a graph-based model only containing the decoder and then copying all weights from the old model to the new one. The resulting checkpoint now contains a graph that can be loaded by the TensorFlow C API.

3.2.2 Custom models

In order to make use of the DDSP training library and extend the synthesizer with additional models, we created four custom models trained on:

- Bass sounds of the Moog One, Moog Minimoog and Moog Minitaur synthesizers
- Studio recordings of Middle Eastern instruments, the Hammered Dulcimer and Santoor
- Studio recordings of a Handpan (also known as Hang Drum)
- Nature field recordings of birds chirping

For training we used the official DDSP (version 0.14.0) Jupyter notebook on Google Colab called *train autoencoder* [6] which allows training on a Google Cloud GPU using own data. According to the recommendations of the DDSP authors given in the notebook, trained models perform best using recordings of a single, monophonic sound source, in one acoustic environment, in .wav or .mp3 format with a total duration of 10 to 20 minutes. Since the DDSP Autoencoder is conditioned on the loudness A and the fundamental frequency f_0, i.e., the model learns to associate different synthesizer configurations to specific value pairs of (A, f_0), training on multiple instruments, acoustic environments or polyphonic sounds prevents the autoencoder to learn a unified representation. Although the recordings listed above are less conform with these training guidelines, we chose them to challenge the DDSP autoencoder, exploring limitations and opportunities in a musical context by deliberately bending the recommended usage.

The training process is performed as follows. The first step is comprised of data generation and pre-processing of the training data. The raw audio is split into short parts of a few seconds, each analyzed on the specified features, i.e., the fundamental frequency and loudness, and finally saved in the TensorFlow *TFRecord* format. The fundamental frequency is thereby estimated by using the state-of-the-art pitch tracking technique, called *CREPE* by Kim et al. [9] that applies a deep convolutional neural network on time-domain audio.

The second step is the actual training, using a Python based configuration framework for dependency injection by Google, called *Gin* [7] . In this way, all available training hyperparameters can be defined in a gin config file that is passed to the training function. The training process does not include any optimization techniques, such as a hyperparameter search or early stopping, the authors just recommend in the code documentation to train for 5,000 to 30,000 steps until a spectral loss of about 4.5-5 is reached for an optimal learning representation without overfitting.

The third and last step was a short evaluation based on resynthesis. Here, a training sample was randomly picked, passed through the autoencoder, and checked if it was perfectly reconstructed based on the learned features.

We successfully conducted training of all four models and validated their performance in the previously mentioned timbre transfer demo. While validation using the DDSP library went smoothly and showed musically interesting results, we ran into issues during inference using the TensorFlow C API within our plugin. We monitored a much higher loudness of the custom models compared to the pre-trained models, resulting in a distorted, clipping sound. Furthermore, we detected a constant harmonic distribution independent of the incoming pitch and loudness while the pre-trained models adapt harmonics and frequency response according to these inputs. The overall experience with the training script provided by the DDSP authors is that it works without problems for standard parameters, but as soon as own hyperparameters within the gin framework are chosen, a lot of side-effects appear. For the mentioned reasons, integrating and possibly adapting the custom-trained models to make them work in the DDSP synthesizer will be a part of future work.

3.2.3 Real-time implementation of the models

The original DDSP implementation synthesizes several frames before processing them into one output. Reading through the DDSP code base, we experienced the number of frames (time steps) to be defined by the size of the input audio and a hop size defined by constants in the gin config file of the selected pre-trained model.

For our real-time implementation we wanted to calculate one frame with a size of the input buffer each time the buffer is ready. Given the static nature of our TensorFlow model implementation we were not able to change the number of time steps on the run. Therefore, we set the number of time steps to one. Each run of the TensorFlow model would then

[5] `https://colab.research.google.com/github/magenta/ddsp/blob/master/ddsp/colab/demos/timbre_transfer.ipynb`, last accessed on 2020-12-15

[6] `https://colab.research.google.com/github/magenta/ddsp/blob/master/ddsp/colab/demos/train_autoencoder.ipynb`, last accessed on 2020-12-15

[7] `https://github.com/google/gin-config`, last accessed on 2020-12-15

return a set of values for one time step, independent of the buffer size.

3.3 Additive synthesizer

The implementation of the additive synthesizer can be found in the `additive.m` MATLAB code file. During the development of the DDSP synthesizer we went from a re-implementation of the DDSP equivalent to an adapted real-time optimized version with additional parameters for high-level control. While the original DDSP library provides two different implementations of the additive synthesis, the harmonic and sinusoidal approach, this work focuses on the harmonic synthesis that models a signal by adding only integer multiples of the fundamental frequency.

In the following, the initial implementation as well as the main modifications in its final state are clarified. As already explained in 2.1, the additive synthesizer models audio using a bank of harmonic sinusoidal oscillators. The synthesis algorithm takes amplitudes, harmonic distribution and fundamental frequencies for a specified number of frames as input and computes the sample-wise audio signal as output. The harmonic distribution provides frame-wise amplitudes of the harmonics. The additive synthesis as implemented in the DDSP library is performed in two main steps: 1) Translation of neural network outputs to the parameter space of the synthesizer controls, and 2) Computing the output signal from synthesizer controls.

For 1), the amplitudes were scaled and the harmonic distribution was scaled, bandlimited (i.e., removing the harmonics that exceed Nyquist frequency) and normalized, while the fundamental frequencies remained unchanged. After retrieving valid synthesizer controls in step 1), the harmonic synthesis is performed. Since the DDSP approach works frame-based while the output needs to be delivered sample-based, the synthesizer controls need to be upsampled. This is done by linearly interpolating the frequency envelopes and windowing the amplitude envelopes by using 50% overlapping Hann windows. Having calculated all controls on a sample basis, the signal can be synthesized by accumulative summation of the corresponding phases, i.e., adding the calculated sinusoids together, sample by sample.

The following changes were made to optimize the algorithm for a real-time application and to add additional high-level control for the synthesis.

- Since the frame-based calculation was computationally too heavy, we adapted the code so that the input is always one frame (equivalent to the buffer size) and all computations are sample-based. Therefore, no resampling or windowing is needed.
- Each time the function is called, the phases of all harmonics are saved and returned along with the signal and added as offset in the next call to avoid artifacts caused by phase jumps.
- In order to be able to optionally introduce non-harmonic partials to the signal, a stretch parameter was added that transforms the distance between the integer multiples while maintaining the fundamental frequency. An additional shift parameter adds the functionality to modify the fundamental frequency from one octave below to one octave above the current pitch in a continuous scale.

3.4 Subtractive synthesizer

This component is responsible for the non-harmonic parts of instrument sounds, such as the audible non-pitched flow of air that accompanies the harmonic part of a flute sound. Our implementation, which can be found in the `subtractive.m` MATLAB code file, generates a frame of random noise and then filters it according to a given frequency response.

The function's parameters are the frame length (number of samples), noise color (see below) and the frequency response, which is given as a vector of N magnitudes $m_0, \ldots, m_{N-1}$, where m_0 corresponds to the DC component and m_i to frequency $f_{\text{nyquist}}/(N-i)$ with $f_{\text{nyquist}} = f_s/2$ and samplerate f_s.

While we started with a direct re-implementation of the DDSP FilteredNoise approach described in 2.2, we made the following adaptations over the course of the project:

- **Simplified filtering**: The DDSP synthesizer processes multiple frames at once. For real-time implementation, we removed the step of calculating the impulse response for each frame and applying a windowing function. Instead, we simply perform a Fourier transform on the generated noise and multiply the result with the filter magnitude response that the model predicted for the single current frame.
- **Noise color**: We provide functionality to shape the frequency distribution of the generated noise. Noise color generally refers to the frequency f being emphasized proportionally to $1/f^\alpha$ for some exponent α [10]. $\alpha < 1$ results in higher frequencies becoming more prominent, while $\alpha > 1$ increases the energy of the lower frequencies. Uniform white noise is achieved by setting $\alpha = 1$.

3.5 Graphical User Interface

After the development of all the features of our synthesizer, we focused our attention on designing an interface with high-level controls for the additive and the subtractive synthesis, the reverb, the modulation and the models. Our process started from a list of all the parameters we wanted to manipulate. We also looked for some inspiration from well-known VST synthesizers, comparing them in terms of usability and trying to understand what their best interaction features were. Later we organized the controls of our synthesizer in different modules and displayed them in a rectangular interface, trying to find a layout that was pleasant but also respectful of the instrument's architecture logic. In table 2, we list all the controls for each module of our synthesizer. Because of the particular choice of a graphic control for the harmonics' amplitude, the team opted for a spectrogram representing the output of our plugin. In this way, the user is able to clearly see which harmonics are being played.

Module	Feature controls
Input selector	MIDI/line selector
Models selector	Violin Flute Saxophone Trumpet Moog Bass (not included) Dulcimer (not included) Handpan (not included) Chirps (not included)
Additive synthesis	Graphic harmonics editor f_0 shift Harmonics stretching Global amplitude
Subtractive synthesis	Noise color Global amplitude
Modulation	Modulation rate Delay control Amount
Reverb	Dry/wet mix Size Glow
Output	Master gain
Spectrogram	Clear visualization of the output

Table 2. List of GUI's features

Once we defined the layout and the parameters that we wanted to control, we moved to the software development in JUCE. In order to customize the appearance of knobs, we used the "Custom LookandFeel" objects while we designed ad hoc images for the buttons and background texture using a vector graphics software. Figure 1 previously presented the GUI of our synthesizer.

3.6 Plugin setup

The synthesizer ended up being built as a standalone executable and a DAW plugin using Steinberg's VST3 format.

Using JUCE's `AudioProcessorValueTreeState` class we are exposing the different controllable parameters to the DAW, allowing control and automation of the plugin. Using this class we will also be able to easily store and read plugin states, enabling generation of presets, though this has not been implemented yet.

The synthesizer is configured to load the models from a given path with subfolders containing the individual models, as well as configuration files containing key-value pairs such as number of harmonics and scaling values.

4. EVALUATION

In order to understand the strengths and weaknesses of our product to improve it, we designed an evaluation strategy for both User Experience (UX) and sound output. Our target users are musicians and music producers. Accordingly, we shared a release of our VST plugin with selected sound engineers, musicians and producers to collect opinions and user insights. Moreover, we designed two different questionnaires and asked participants to evaluate the UX and the sound accuracy of our software. The DDSP Synthesizer as well as the two questionnaires have been distributed online and the participants received an email with all the indications to properly conduct the test.

In the following, we mainly describe the UX evaluation, including our approach, desired outcome, survey design and results.

4.1 User Experience Evaluation

4.1.1 Approach

The aim of this evaluation was to collect feedback about the user interface from people with experience on synthesizers and music production. One of the goals of our project was to design a simple and efficient interface able to control several parameters with a single gesture without giving up functionality in the pursuit of simplicity. After a trial period where the participants had the chance to familiarize themselves with the software, we asked them to complete a survey.

4.1.2 Survey structure

We designed the survey with different sections to group the questions by theme. We included an experiment in order to ask each participant to load and perform some changes to a model and export the result in an audio file. In this way, we ensured that every participant had at least used and interacted with the plugin for a while. Moreover we are able to compare each audio export to understand if some of the instructions were not clear or if the UX itself was not effective.

Four usage questions have been asked to collect information about the user's DAW and for how much time they used the plugin. In the next sections we asked the participants to report their experience during the experiment and evaluate the user interface rating 9 different statements with a Likert-scale, a widely used bipolar symmetric scaling method in questionnaires. In this way, users were able to express their agreement/disagreement related to each sentence. Furthermore, we asked 4 open questions to let the participants express their opinion about the overall UX. Finally we added 8 questions to locate demographics and musical-related personal experiences. Table 3 summarizes the content of each section.

#	Section	Content
1	Introduction	Aim of the questionnaire
2	Experiment	Task instructions
3	Usage	4 mixed questions
4	UX evaluation	9 Likert scale evaluations
5	UX experience	4 open questions
5	Demographics	8 mixed questions

Table 3. Content of the UX survey

4.1.3 Expected results

Considering that the software was still under development, we were expecting reports about compatibility issues with different DAWs as well as some stability problems. Moreover, because of the VST's instability in the first release, it is possible that some users will not be able to conduct the small experiment that requires the plugin to be embedded in a DAW track. Considering the whole interface, one of the main points of our design requirements was the simplicity and thus our hope is to facilitate the user's interaction. Even if the number of participants is limited, we expect that the users will approximately identify 75% of the UX issues accordingly to Nielsen's model [11].

4.1.4 Results

We received seven answers. Five participants identified as males, one female and one preferred not to say. The age average is 28.57 years (STD 8.42). Six of them declared that sound production is their hobby while one said music production is related to their job. The mean experience in the music production field is 7.43 years (STD 4.87). Six users do not have experience with machine learning VST plugins and only one of them does not know if she/he ever used one. Each user spent an average of 23.57 minutes using our synthesizer (STD 17.74). We suppose that some mistake has been made reporting the usage time for at least one user. In table 4 we report the number of user tests per different software environment.

# users	**Environment**
3	Reaper
2	Ableton Live
1	Cubase
1	Standalone version

Table 4. List of used DAWs in the evaluation.

In general, the experiment has been rated a medium difficult task with a mean rating of 3.43 in a scale from 1 to 5 being 1 "easy to accomplish" and 5 "hard to accomplish". In figure 3 we summarize the answers obtained from the questions with an associated Likert scale. The users were asked to rate each sentence from 1 to 5 with 1 corresponding to "strongly disagree" and 5 to "strongly agree". We can observe that the graphical user interface has been really appreciated with a 4.43 mean value while the interface's controls seem not to let the participants easily reach the wanted results. The other statements reported in the Likert section obtained a medium rating between 3 and 3.86 which might mean that the GUI is in general appreciated.

As expected, some of the participants encountered difficulties in the installation procedure of the VST3 plugin in both Windows and macOS environments while the standalone version seems to be more stable. Furthermore, three users reported an unsatisfactory audio result related to the presets obtained from models. Here we report part of one of the feedback: "*[...] It's possible to get some cool sounds but the default sound when you just start it is not so nice.*". On the other hand, the audio input feature was appreciated: "*[...] I think the audio input feature has a lot of potential and I caught myself experimenting with this a lot and loosing track of time.*". Two participants reported that the possible interaction with the interface for the additive synthesizer was not immediate to spot and they realized its features after a while. For this reason they suggest a graphical indication to guide the user to the interaction with the harmonic sliders. A significant outcome is the unexpected audio results that participants reported. Even though they described output sounds as "awkward", they highlighted the new creative way of producing unexpected sounds, finding the whole synthesizer experience engaging.

4.2 Real-time timbre transfer

Running DDSP decoder models in a real-time plugin is computationally feasible. As the demonstration [8] shows, the plugin does not exceed 20% CPU load on an AMD Ryzen 7 with a clock speed of 2.9 GHz. Similar results were measured on a MacBook Pro 2013, with a 2GHz Core i7 processor.

We found the quality of the timbre transfer in our real-time implementation below that of the demonstrations published by the Magenta team. Our converted models preserve some characteristics of the original ones, such as wind noises in the flute model, but do not accurately reproduce the timbre overall. We confirmed that on the level of a single frame, our models produce the same output as their original counterparts; will investigate and improve the quality in the future. Additionally, we would like to further investigate why we were unable to perform the timbre transfer with models that we trained both within the framework provided by Magenta, and within custom environments.

4.3 Distribution as a VST3 plugin

When it came to distributing our project to users, we encountered some difficulties in packaging the required libraries and model files together with the generated VST3 plugin. Some of the DAWs that users tested on, like Ableton or Reaper, did not recognize the plugin or experienced stability issues during its usage. Although the core functionality could still be accessed via the standalone application generated by JUCE, the project was designed first and foremost as a plugin. Functionality like handling of external audio sources and wet/dry mixing was expected to be handled by the host DAW. Users who had to resort to the standalone when their DAW did not recognize or stably run the plugin reported those features as missing.

Thus, we would like to improve the distribution process in the future, ensuring that the project can be seamlessly installed as a plugin in multiple DAWs on Windows and macOS.

5. CONCLUSION

In this paper, we presented an approach to integrate the DDSP library into a real-time plugin and standalone application using the JUCE framework. We succeeded in

[8] `https://share.descript.com/view/hXAZLCPJNqm`

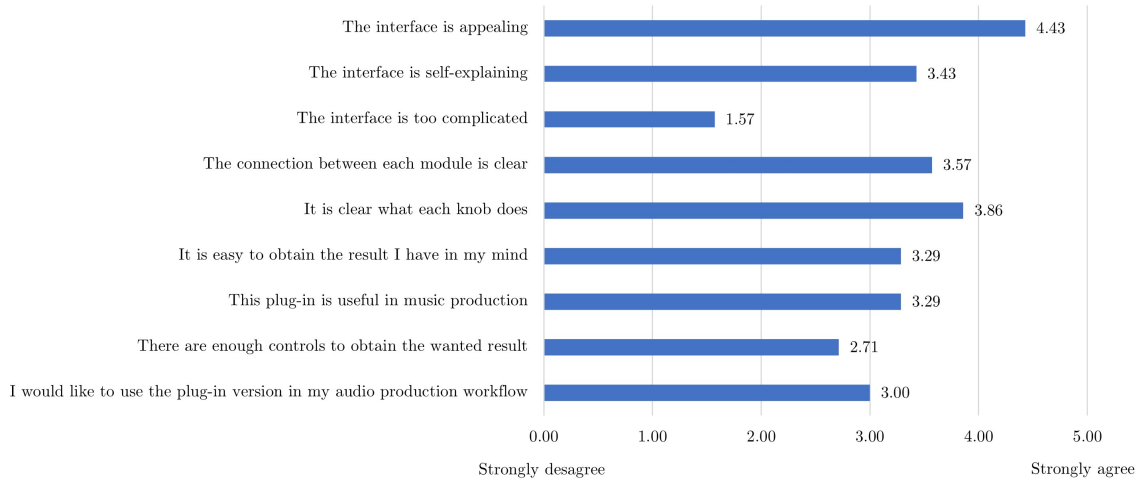


Figure 3. User experience evaluation - Likert scale

implementing a synthesizer playable based on pure user input. While we were generally able to use the output from pre-trained models to control the DDSP backend, further research is needed to match the sound quality of these real-time models to that of the offline timbre transfer examples provided by the DDSP authors.

A recently released realtime reimplementation of DDSP in PyTorch [9] provides a possibly more seamless way of interfacing with DDSP models in C++ that proved compatible with our plugin and JUCE. Extending that API to allow the user some control over the synthesis parameters seems a promising avenue to improve the sound quality of our timbre transfer.

6. REFERENCES

[1] C. Donahue, J. McAuley, and M. Puckette, "Adversarial Audio Synthesis," *arXiv:1802.04208 [cs]*, Feb. 2019, arXiv: 1802.04208. [Online]. Available: http://arxiv.org/abs/1802.04208

[2] M. Blaauw and J. Bonada, "A neural parametric singing synthesizer modeling timbre and expression from natural songs," *Applied Sciences*, vol. 7, p. 1313, 2017.

[3] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," *International Conference on Learning Representations*, 2020.

[4] X. Serra and J. O. Smith, "Spectral modeling synthesis. A sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

[5] Native-Instruments, "Razor," 2011. [Online]. Available: https://www.native-instruments.com/en/products/komplete/synths/razor/

[6] D. Pandey, U. Suman, and A. Ramani, "An effective requirement engineering process model for software development and requirements management," in *2010 International Conference on Advances in Recent Technologies in Communication and Computing*, 2010, pp. 287 – 291.

[7] A. Cheveigné and H. Kawahara, "YIN, A fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, pp. 1917–30, 2002.

[8] P. Brossier, "Automatic annotation of musical audio for interactive applications," Ph.D. dissertation, Queen Mary University of London, 2006.

[9] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "CREPE: A convolutional representation for pitch estimation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 161–165.

[10] N. J. Kasdin, "Discrete simulation of colored noise and stochastic processes and $1/f^{\alpha}$ power law noise generation," *Proceedings of the IEEE*, vol. 83, no. 5, pp. 802–827, 1995.

[11] J. Nielsen and R. Molich, "Heuristic evaluation of user interfaces," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 1990, pp. 249–256.

[9] https://github.com/acids-ircam/ddsp_pytorch

EXPLORATIONS OF SINGING VOICE SYNTHESIS USING DDSP

Juan ALONSO (jalons19@student.aau.dk) [1] and **Cumhur ERKUT** (cer@create.aau.dk) (0000-0003-0750-1919) [1]

[1] *Sound and Music Computing, Department of Architecture, Design, and Media Technology*, **Aalborg University**, A.C. Meyers Vænge, Copenhagen, DK-2450 Denmark

ABSTRACT

Machine learning based singing voice models require large datasets and lengthy training times. In this work we present a lightweight architecture, based on the Differentiable Digital Signal Processing (DDSP) library, that is able to output song-like utterances conditioned only on pitch and amplitude, after twelve hours of training using small datasets of unprocessed audio. The results are promising, as both the melody and the singer's voice are recognizable. In addition, we explore the unused latent-z vector in DDSP to improve the lyrics. Furthermore, we present two zero-configuration tools to train new models, including our experimental models. Our results indicate that the latent-z improves both the identification of the singer as well as the comprehension of the lyrics.

1. INTRODUCTION

Human voice is one of the oldest musical instruments [1]. Before the Deep Learning era [2], high-quality singing synthesis was carried out either by the spectral models [3], based on perception, or the physical models [4], based on production and articulation. Combining the spectral models with deep learning, the Differentiable Digital Signal Processing [5] (DDSP) library by Google's Magenta team became a powerful toolkit for audio-related machine learning. The first published examples of DDSP were focused on timbre transfer from monophonic instruments.

In this paper we present the DDSP architecture and apply it to a more complex, expressive instrument: the human voice. We check the suitability of the DDSP for singing voice synthesis. By constructing small size databases, experimenting with the model parameters and configurations, and by training the resulting models only for about twelve hours, we obtain singing-like outputs, which clearly resemble to original singers / speakers. However, the lyrics are incomprehensible because we don't have a language model. When we condition the model on the MFCC of the original audio, the results improve, promising intelligibility. Our contribution also enhances the documentation of the library and provides two zero-configuration notebooks for experimentation.

This paper is organized as follows. In Sec 2, we introduce the context of neural singing synthesis. Next we provide a detailed look at the DDSP architecture for timbre transfer. In Sec. 4, we introduce our experiments together with the data sets and model configurations, and the improved results we have obtained by adding the latent-z vector. In the next section, we discuss our observations. We finally draw our conclusions and indicate areas of further research.

2. BACKGROUND

In neural singing synthesis, the Deep Neural Network (DNN) receives a list of pitches and the lyrics as input, and outputs a signal modeling a specific voice. It is a problem closely related to the speech synthesis, however more challenging because of more diverse sets of pitches and intensities, different vowel durations, and other attributes specific to singing.

Gómez et al. [6] revised many data-driven deep learning models for singing synthesis. A thoughtful remark in that paper is that the black-box characteristics of deep learning models make it very difficult to gain knowledge related to the acoustics and expressiveness of singing. Even if deep learning techniques are general and can learn from almost any arbitrary corpus, it is necessary to advocate for explainable models to break the black-box paradigm.

The DDSP library is a set of tools released by Google's Magenta team. DDSP is set to bring explainability and modularity in neural audio synthesis [5]. The idea behind DDSP is "*to combine the interpretable structure of classical DSP elements (such as filters, oscillators, reverberation, envelopes...) with the expressivity of deep learning.*" [1] To avoid a common misunderstanding, DDSP is not an architecture *per se*, but a set of signal-processing tools that can be incorporated into modern automatic differentiation software. Many examples of DDSP relate to singing input, therefore we provide an overview in this section.

Several papers extended the DDSP specifically for speech and singing synthesis [7–9]. In [7], the model is conditioned on the phoneme level using an Automatic Speech Recognition (ASR) system to extract the phonemes of the training set and use them as additional conditioning data. In [8], the authors synthesize spoken speech using the DDSP architecture with a model conditioned on mel-spectrograms, instead of using raw audio. The loss function is also adapted to use mel-spectrograms trying to mimic the way human perception works. In [9], the authors propose using MIDI data modified by an LSTM network, to obtain continuous pitch and loudness contours that will be fed into the DDSP architecture.

[1] https://magenta.tensorflow.org/ddsp

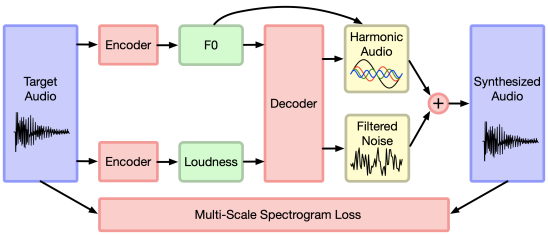


Figure 1. Timbre transfer architecture used in this work. Adapted from [5].

2.1 DDSP Overview

Phase alignment poses a problem when generating audio in deep learning; that problem is present when frames are used, either in the time- or the frequency-domain. An autoregressive model does not present this problem, but instead is harder to train due to the amount of data needed, and due to the interplay between audio perception, wave shape and loss. In fact, two wave shapes with very different losses can be perceived as sounding exactly the same.

One possible solution is the use of audio oscillators or vocoders that perform analysis and synthesis. In analysis, interpretable parameters such as f0 or loudness are extracted, and in synthesis the generative algorithm uses these parameters to construct the synthetic sound. DDSP disentangles them by a series of classical Digital Signal Processing (DSP) modules as feed-forward functions allowing, efficient implementation on GPUs.

3. DDSP AND TIMBRE TRANSFER

The architecture used in this work is based on the "solo_instrument" setup proposed for timbre transfer in [5] and shown in Fig 1. The network is presented an audio file. f0 and loudness are extracted and fed to the decoder, which produces the parameters for a harmonic synthesizer and for a subtractive synthesizer. The audio from both synthesizers is combined and presented as the final output. During the training phase, the loss is computed using different resolution spectrograms from the original and the resulting signal. In the following, we describe the modules used.

The **encoder** transforms the incoming audio into latent vectors, in this case, two interpretable features: the fundamental frequency and the perceived loudness of the monophonic input audio. The DDSP library does not require a specific method to generate the **f0** and **loudness** vector, but expects arrays of float values, sampled at 250 Hz.

f0 can be generated synthetically, but the DDSP library includes examples using CREPE [10] and DDSP-inv [11]. The f0 latent vector is fed directly to the additive synthesizer. This allows to disentangle the fundamental frequency and facilitates the model to respond to frequencies unseen during the training.

Loudness can also be generated synthetically, but the DDSP library includes a utility function to compute the perceptual loudness of the audio, applying A-weighting curves to the power spectrogram.

The **decoder** (Figure 2, top) is an RNN that receives the latent vectors (f0 and loudness) and outputs the control parameters required by the synthesizers: the amplitudes of the harmonics, and the transfer function for the FIR filter. The RNN is fairly generic, as the DDSP authors emphasize that the quality of the results comes from the DDSP modules, not from the complexity of the neural network.

The latent vectors are preprocessed by a Multilayer Perceptron (MLP) (Fig. 2, bottom), which comprises a block of three layers (Dense, Normalization and ReLU layers) repeated three times. The output of the MLP is connected to a 512-cell GRU layer which is connected to a second MLP with the same structure and finally passed to two dense layers that will provide the parameters for the synthesizers.

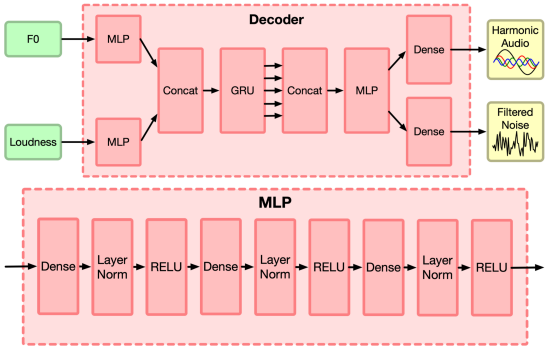


Figure 2. Decoder architecture (top) and MLP structure (bottom). Adapted from [5].

To synthesize audio, the DDSP library uses **Spectral Modelling Synthesis** (SMS), a technique proposed by Serra and Smith [3], where the sound is modeled as two components: an additive or harmonic synthesizer, where a series of sinusoids is combined, and a subtractive synthesizer where the residual component is modeled as filtered noise. The DDSP library implements a differentiable version of the SMS, with an additional constraint: all the frequencies are integer multiples of f0. The expressiveness of this technique is a consequence of the high number of parameters needed. With the default configuration (60 harmonics, 65 noise magnitudes and 1 amplitude), sampled at 250Hz, 1 second of audio yields (60+65+1)*250 = 31,500 dimensions vs 16,000 audio samples.

The **additive (harmonic) synthesizer** models the harmonic part of the signal $x(n)$ as a sum of K sinusoids, with amplitudes $A_k(n)$ and instantaneous phases $\phi_k(n)$.

$$x(n) = \sum_{k=1}^{K} A_k(n) \sin(\phi_k(n)). \quad (1)$$

We can further expand $A_k(n)$ into a global amplitude $A(n)$ and a normalized distribution $c(n)$ of amplitudes for the harmonic components, so that $A_k(n) = A(n)c_k(n)$. Then Equation (1) can be rewritten as

$$x(n) = A(n) \sum_{k=1}^{K} c_k(n) \sin(\phi_k(n)), \quad (2)$$

The instantaneous phase is described as

$$\phi_k(n) = 2\pi \sum_{m=0}^{n} k f_0(m) + \phi_{0,k} \quad (3)$$

where $\phi_{0,k}$ is the initial phase for the harmonic k.

To reconstruct the additive signal at audio sample rate, f0 is upsampled using bilinear interpolation, and the amplitude and harmonic distribution are smoothed using an overlapping Hamming window centered on each frame.

The **subtractive synthesizer** models the residual component, the difference between the original signal and the signal from the additive synthesizer. If we assume that the residual component is stochastic, it can be modeled as filtered white noise, using a time-varying filter. The filter is applied in the frequency-domain, to avoid phase distortion. For every frame l, the network outputs H_l, the frequency-domain transfer function of the FIR filter. The convolution is applied as a multiplication in the frequency-domain (Eq. 4), and then an Inverse Discrete Fourier Transform (IDFT) is applied to recover the filtered signal (Eq. 5):

$$Y_l = H_l DFT(x_l), \quad (4)$$

$$y_l = IDFT(Y_l). \quad (5)$$

Training the autoencoder means finding a set of parameters for the synthesizers that minimize the reconstruction loss i.e., minimize the difference between the output and input signals. A sample-wise loss measure is not recommended, as two similar waveforms can be perceived as having a very different sound. The spectral loss L –the difference between the spectrograms of the input (S) and output ($\hat{S}$) signals– is perceptually better, but there is a compromise between time and frequency.

To solve this problem, a multi-scale spectral loss is defined. Instead of using a single pair of spectrograms $(S, \hat{S})$, the loss is defined as the sum of different spectral losses L_i where i is the FFT size. Moreover, the linear magnitude losses are sensitive to the peaks, whereas logarithmic magnitude losses are sensitive to the quiet regions of the signals. Therefore the loss is defined as $L = \sum_i L_i$, with $i \in (2048, 1024, 512, 256, 128, 64)$, where

$$L_i = ||S_i - \hat{S}_i||_1 + ||log(S_i) - log(\hat{S}_i)||_1 \quad (6)$$

4. EXPERIMENTS AND RESULTS

Our first experiment [2] is a simple test to check if the system is correctly set up. The second experiment tries to determine if the model is able to learn single from single- and multiple-voice datasets. The third experiment explores the effect of training the model using the same dataset, with different sets of parameters for the spectral synthesizer. The last experiment introduces a encoded representation of the Mel Frequency Cepstrum Coefficients. Tables 1, 2 and 3 describe the datasets we used.

[2] Our notebooks and configuration files are available at `https://github.com/juanalonso/DDSP-singing-experiments`. Example results (audio and spectrograms) are at `https://juanalonso.github.io/DDSP-singing-experiments/`.

Name	Speaker / Singer	Gender	Type	Language
alba	Alba	Female	Spoken	Spa
mrallsop	Scott Allsop	Male	Spoken	Eng
eva	Eva Páez	Female	Sung	Eng, Spa
belen	Belén Chanes	Female	Sung	Spa
servando	Servando Carballar	Male	Sung	Spa
voices2	belen + eva	Female	Sung	Eng, Spa
voices3	belen + eva + servando	Mixed	Sung	Eng, Spa
felipe_vi	King Felipe VI	Male	Spoken	Spa

Table 1. Dataset metadata.

Name	Source
alba	Scrapped from `https://albalearning.com/`
mrallsop	Scrapped from `https://tinyurl.com/mrallsop`
eva	Provided by singer
belen	Provided by singer
servando	Provided by singer
voices2	Provided by singers
voices3	Provided by singers
felipe_vi	Scrapped from `https://tinyurl.com/felipe-vi`

Table 2. Dataset sources.

To test the model, we used a fragment of 'Over the rainbow' as sung by Lamtharn (Hanoi) Hantrakul as the original audio presented to the model. This melody is also used in the DDSP [5] examples.

4.1 Pre-evaluation

A quick test of the system has been carried out to reproduce the timbre transfer experiments while checking the validity of our setup. In this test we have used the same material as in the original paper [3] , generating the dataset and training the model for 30k steps using the standard configuration file `solo_instrument.gin` (60 sine waves, 65 noise magnitudes, reverb and batch size of 16).

The estimated f0 is transposed up two octaves, to better match the pitch range of the violin, and the loudness is attenuated 20dB. Two reconstructions are produced, the first one with no additional information, and the second one using masking and statistical amplitude adjustment. These reconstructions are shown in Fig. 3.

[3] Movements II, III, IV, VI and VIII from the Bach Violin Partita no.1 BWM 1002, as played by John Garner at `https://musopen.org/music/13574-violin-partita-no-1-bwv-1002/`

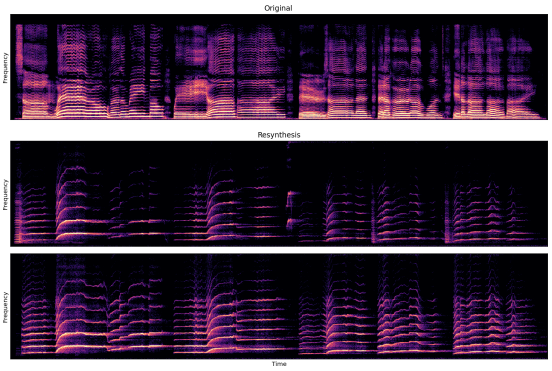


Figure 3. Over the rainbow as sung by Lamtharn (Hanoi) Hantrakul (top) and reconstructed by our version of the violin model (+2oct, -20dB), using no statistical information (middle) and with statistical information (bottom).

Name	Avg. pitch	Duration	Active audio
alba	53.76	1:24:29	0:53:28
mrallsop	48.42	0:25:41	0:23:51
eva	64.10	0:18:34	0:16:23
belen	62.39	0:10:16	0:06:24
servando	55.08	0:11:02	0:08:31
voices2	63.59	0:28:50	0:22:47
voices3	61.75	0:39:52	0:31:17
felipe_vi	50.93	0:13:32	0:08:35

Table 3. Audio properties of the datasets. Active audio is the duration after removing parts of the original file where audio is lower than -52dB for more than 200ms. This is only a reference value and it is not used for training.

The violin model works as expected. The resulting audio is equivalent to the audio produced by the original model, which is trained for 20k extra steps with a batch size of 32.

4.2 Different datasets trained on the same parameters

4.2.1 Single voice model

To generate the models for singing voices, we are using seven datasets (see Table 1), obtained from speakers and singers, both male and female, in different languages. No audio source separation software is used. The datasets eva, belen and servando are raw vocal tracks recorded by professional singers and have been kindly provided by the performers upon request of the authors. These tracks were not specifically recorded for this work, they were previously recorded for existing and upcoming music records. Two additional datasets have been created by combining the audio files from the eva and belen datasets (voices2) and the eva, belen and servando datasets (voices3). The rest of the datasets (mrallsop, alba and felipe_vi) has been scrapped form the web. Files longer than five minutes have been split into three to four-minute chunks. Other than that, the original audio has not been transformed in any way, keeping all the original characteristics: silences, different volume levels, background noise, etc.

The models are trained for 40k steps each, using the notebook `01_train`. Each model has been trained using `singing.gin`, a modified version of the standard configuration file. The losses after training (Fig. 4, top) are all in the range [4.987, 5.437] as recommended in [5]. There are no significant differences between the losses in the spoken and sung datasets. The servando model, whose loss is considerably higher, is an exception. The only apparent difference with the rest of datasets is that servando's source audio is hard-clipped / compressed at 0dB, with a smaller dynamic range than the other voices, which present a wider range of amplitudes and compression values.

Name	Gender	Type	Loss
servando	male	sung lyrics	6.984
belen	female	sung lyrics	5.114
eva	female	sung lyrics	4.987
mrallsop	male	spoken speech	5.437
alba	female	spoken speech	5.142
voices2	female	sung lyrics	5.415
voices3	mixed	sung lyrics	6.143

Table 4. List of datasets used for training the model and loss value after 40k steps.

The output is generated by executing notebook `02_run`. The pitch shift is chosen manually by comparing the mean MIDI pitch of the dataset with the mean MIDI pitch of the melody. The loudness shift is handpicked after comparing different settings. The threshold and quiet parameters are adjusted manually depending on how much noise bleeds into the silent parts of the original audio. The values chosen for each example are shown in the audio web page.

4.2.2 Multiple voices model

The model **voices2** combines the source audio from belen and eva, both female singers. It is trained for 40k steps. The loss, after training is 5.415, higher than the loss of both of the datasets (belen=5.114, eva=4.987), as shown in Fig. 4, bottom. This result is expected, as we are training the model with the same number of steps as the single voice models but, in this case, the model needs to learn two different timbres, with different loudness and slightly different MIDI mean pitches (belen=62.39, eva=64.10).

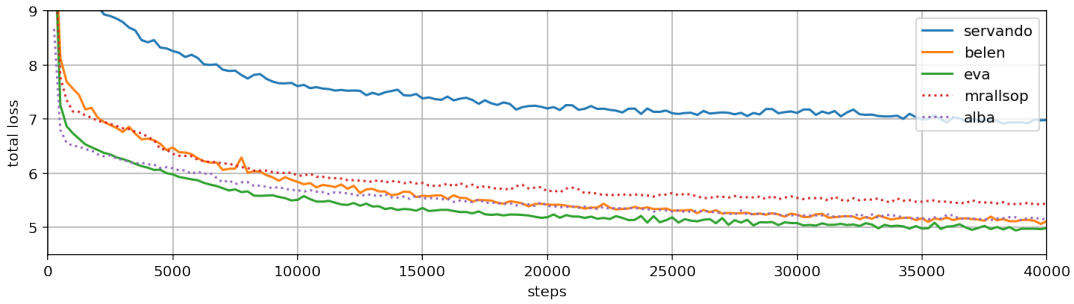
When the model is used, the resulting audio is a combination of the individual voices. Depending on the f0 and loudness, the model outputs a succession of fragments by the original singers. The transition is smooth: only one voice is perceived at a concrete frame.

The model **voices3** combines the source audio from belen, eva and servando. It is trained for 40k steps: the model must learn three timbres, and one of them is more different from the other two (servando, loss=6.984, MIDI mean pitch=55.08) The peculiarities of the servando dataset penalize the training, and thus the model presents a higher loss than voices2.

Fig. 4, bottom, shows that the loss of voices3 (6.143) is lower than the loss of the servando model (6.984). We attribute this effect to an imbalanced dataset: the duration of servando's source audio is 11 minutes and 02 seconds, whereas belen's and eva's source audio combined is 28 minutes and 50 seconds.

In this experiment, the voice mixing capabilities of the model are more pronounced than in the previous experiment. The mean MIDI pitch of the example song is 51.30. Considering that the mean MIDI pitches of the servando, belen and eva datasets on Table 4 are, respectively 55.08, 62.39 and 64.10, we can expect that when rendering the audio, the model will generate a mix of the nearest-pitched voices. This is the case: when using the example fragment without transposing, the resulting melody is a mix of servando's and belen's voice. If the example fragment is transposed an octave higher (MIDI pitch of 63.30) the resulting melody is a mix of belen's and eva's voice. To demonstrate this effect, six examples have been uploaded to the audio page, using different sets of preprocessing parameters.

(a) Five independent datasets: dotted lines represent speech datasets.



(b) Hybrid dataset, with voices2 and voices3.

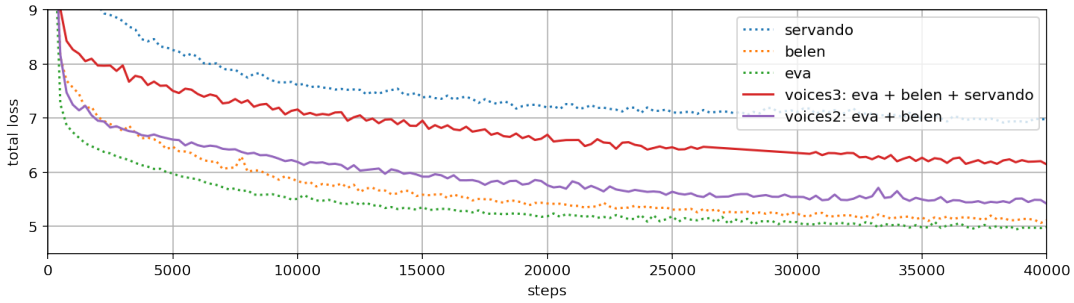


Figure 4. Spectral loss for the different datasets

Use statistics	Yes
Mask threshold	1
Quiet	20
Autotune	0
Octave shift	+1
Loudness shift	-10dB

Table 5. f0 and loudness preprocessing parameters.

4.3 Single dataset, different spectral parameters

In this experiment, we want to understand how the parameters of the spectral synthesizer affect the learning process and the results. We will be using the eva dataset –since its model has the lowest loss after training– to train the model using a range of spectral configurations.

We chose three values for the harmonic component (20, 60 and 100 sinusoidal waves). 60 is the default value provided in the configuration files. 100 is the maximum value a model can be trained without getting out of memory errors, and 20 is the symmetrical lower bound ($60 - (100 - 60)$). For the noise component we chose 10, 35 and 65 noise magnitudes, 65 being the default value.

Nine models are generated, one for each combination of harmonic components and noise magnitudes. Each model is trained for 20k steps, using the same configuration file we used in the previous sections (`singing.gin`). f0 and loudness are preprocessed using the handpicked parameters from Section 4.2.1 and shown in Table 5.

As can be observed in Fig. 5, to decrease the spectral loss, the amount of noise magnitudes is more relevant than the number of harmonic components. Perceptually, more models and tests are needed. On an informal test where three users with no musical training were presented pairs of the snippet 'way up high' (seconds 7 to 11 in the original audio) rendered with different parameters, there was no agreement on which "sounded better". The only exception was snippet h:20 n:10, where the subjects remarked it was the worst sounding of the pair. All subjects commented on listening fatigue due to being exposed to the same melody.

Observing the spectrograms of the reconstructions in Fig. 6, the examples with the lowest number of noise magnitudes ($n = 10$) show the model trying to reconstruct the high frequency noise with the harmonic model (faint sinusoids in the spectrograms).

4.4 Adding the latent-z

For our last experiment, we expand the `singing.gin` configuration (Fig. 7) to include a time-varying encoding of the Mel Frequency Cepstral Coefficients (MFCC) to understand how this additional encoding affects the model's output. This representation is specially well suited for the human voice, as it mimics the non-linearity of human sound perception and helps model the timbre [12]. The MFCC are encoded using a normalization layer and a 512-unit GRU, reducing the dimensionality from 30 coefficients to a 16-dimensional latent vector, at 125 frames per second and then upsampled to 250 frames, to match the sampling rate of the f0 and loudness vectors. To decode this vector, the same MLP architecture shown in Fig. 2, bottom, is used, and concatenated with the preprocessed f0 and loudness vectors.

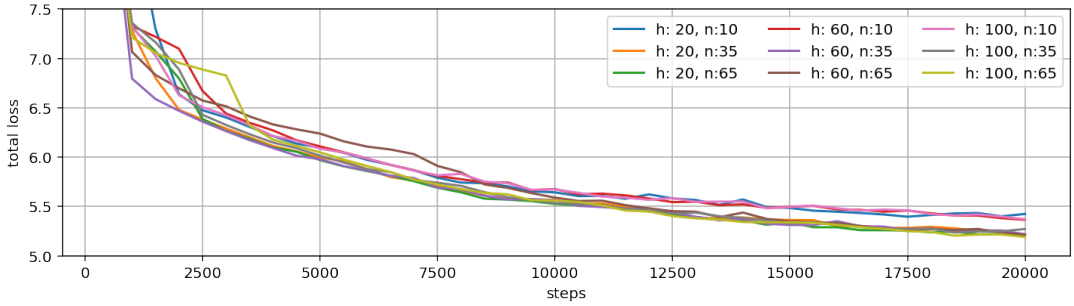


Figure 5. Total (spectral) loss for the eva dataset, using different parameters for the spectral synthesizer. h is the number of harmonic components; n is the number of noise magnitudes.

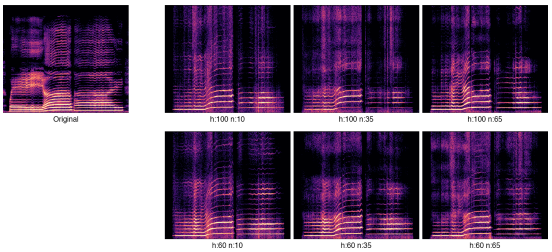


Figure 6. Spectrogram of the original audio ('way up high', seconds 7-11). left column, and spectrograms of the model's output with different spectral parameters.

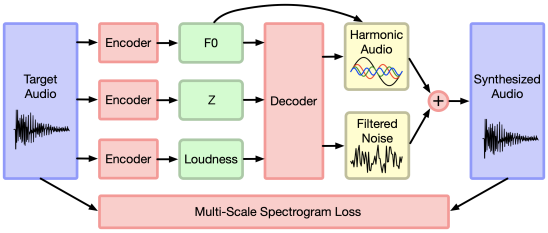


Figure 7. Timbre transfer architecture with latent-z. Adapted from [5].

To generate the z models, we used the belen and mrallsop datasets and a new dataset, felipe_vi, extracted from Felipe VI's, King of Spain, 2020 Christmas speech (male spoken voice, Spanish) As with the previous datasets, felipe_vi is used "as is", without any kind of preprocessing. A new configuration file (`singing_z.gin`) is also used, and it is available at the GitHub repository. This configuration file inherits all the values from `singing.gin` and includes the z encoder.

The models are trained for 25k steps. As shown in Figure 8, with the additional encoding, the loss function increases (from 5.310 to 6.322 in the case of the belen dataset and from 5.634 to 8.011 in the mrallsop dataset) This is an expected behavior, as the model now needs to fit additional parameters. The loss value for the felipe_vi_z model is low enough (5.514) to be in the range of the non-z models.

5. RESULTS AND DISCUSSION

The architecture proposed in [5] is powerful enough to perform timbre transfer and to produce surprising results even if the dataset is small (10-15 minutes of audio). The training, using the Colab environment, is fast and it works out of the box, with the GPU infrastructure ready. All the required libraries (CREPE, TensorFlow, Apache Beam. etc.) are available without version conflicts.

In singing voice synthesis, we challenged the model (small datasets, unprocessed audio, etc.), but the quality of the results surprised us. Of course, in no way the output of the model is going to be mistaken for a human, but the model's ability to produce such good results with no additional conditioning is very promising and opens several avenues for exploration, research and creative usage.

With the addition of the z-encoder, the quality of produced audio is increased, becoming almost intelligible. The right vowels start appearing, and the babbling effect is reduced substantially. The resulting timbre is halfway between the instrument timbre and the original timbre.

This architecture makes very difficult to estimate the performance of a model. As we have noticed, the training loss of the servando model is quite high, compared with the rest, but when analyzing the dataset, nothing stands out as the cause of this value. Similar datasets (speaking, male voice) such as felipe_vi_z and mrallsop_z present very different losses (5.514 versus 8.011 respectively), but the quality of the resulting audio is comparable.

5.1 Problems

5.1.1 Babbling and stuttering

We are forcing the model to recreate sung lyrics. The model needs to learn the timbre of the voice, how to extrapolate previously unheard pitches and the flow of the language. The current architecture manages to extract the timbre and to correlate f0 and loudness with sounds, but it lacks the ability to learn the sequences of phonemes that constitute speech. Even with more comprehensive datasets, where all the possible combinations of phonemes and pitches could be present, without additional conditioning (phonetic, text, etc.) the model will try to make up what to say, and the produced audio will be just a better-quality

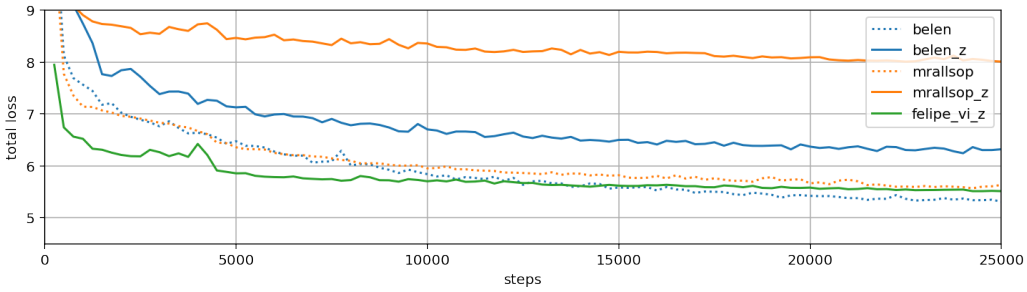


Figure 8. Spectral loss for the three z models. The dotted lines are included for reference, and represent loss function for the two related non-z models.

version of this stuttering and nonsensical babbling.

During the development of this work, the audio has been presented to several listeners who knew the original singers (belen, servando) and they all found the results unsettling, due to this kind of babbling. They recognized the speaker, but the babbling and stuttering were compared to listening to a person having suffered a stroke that impaired the language centers of the brain.

5.1.2 Silence

If the dataset does not include silences (for example, the dataset used to train the violin model) the resulting model has difficulties trying to recreate them and will resort to generate some very low notes. This can be mitigated by adding some transitions to and from silence and by fine tuning the preprocessing parameters, which right now it is a manual process dependent on the input audio and the model. The datasets used in this work do not present this problem, since the original material includes pauses and therefore, the network can recreate possible silences.

The example on the audio page shows this phenomenon particularly well. On the one hand, the original audio, a staccato synthesizer riff, is played as legato. On the other hand, the silence that occurs at seconds 3 and 9 is reinterpreted as a pair of low-pitched tones. Even tweaking the preprocessing parameters, we can mitigate the low tones, but not suppress them.

5.1.3 Pitch artifacts

The accuracy of the output pitches depends on the f0 estimation. If the estimation made by the CREPE model is wrong, the output will include wrong notes. We have detected several cases where CREPE estimates a very high pitch with high confidence, so the preprocessor cannot mask it. In those cases, the resulting audio will include a squeal, a quick glissando to the estimated high pitch.

To avoid this, we can substitute CREPE for another algorithm, or use a symbolic notation, such as MIDI, to generate the f0 vector. In that case, we risk having a monotonous voice, and we would need to add some modulation (e.g., amplitude or frequency modulation) to make it more natural sounding.

5.1.4 DDSP maturity

Although the possibilities of these libraries are immense, exploring them is a challenging task for two major reasons. The first one is the lack of information about the most complex processes, how some of its modules work and how to modify the workflow to add new functionalities. Despite open-sourcing the code in GitHub, the tutorials and demos barely scratch the surface. The second problem is that, because the libraries are still under active development, some of the updates are missing information about the release changes and are not as stable as expected. These, however are expected in any open source library.

6. CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

The DDSP library opens up a lot of possibilities for audio synthesis. The work presented here allows us to get a better understanding on how the DDSP library works, especially when used for timbre transfer. It achieves two goals:

1. **Test the validity of the DDSP architecture to generate a singing voice.** The tests carried out on the architecture have used unfavorable data (no preprocessing, background noises, small datasets, etc.), and even so, the network could generate audio similar to the human voice, with enough features to be recognized as belonging to a specific singer.

2. **Create an easy-to-use environment to facilitate model training and timbre transfer to end users**. The notebooks provided will help the SMC community to ease the learning curve of this architecture and get familiar with the advantages and nuisances of the library. Since the GitHub repository includes some of the models used in this work, curious users can just interact with them, without needing to create their own datasets. Also, as per today (March 2021) our models are compatible with the ones made available by Google (flute, violin...) so they can be used interchangeably.

6.2 Future Work

In order to create a more refined model which is capable of synthesizing much realistic utterances with lyrics replication (and thus avoiding the gibberish / stuttering effect) additional work must be done in the following areas:

Conditioning: As noted in Section 5.1.1, the phonetic output is made up by the model, without any reference to real speech. The current architecture does not condition the output in any other data than pitch and loudness, missing additional information present in sung lyrics. To get the nuances of human singing and model the lyrics, we need to include additional conditioning on the language level, for example, the phonetic conditioning proposed in [7].

Use representations more suitable to voice synthesis: The default architecture proposed in the DDSP is generic, designed for monophonic musical instruments. Using mel-spectrograms as proposed in [8], instead of using raw audio or by postprocessing the harmonic and the noise components of the transformed audio to balance the voiced and unvoiced parts of the speech [7], results could be improved.

Use synthesizers more suitable to voice modeling: As stated previously, by using Spectral Modelling Synthesis, we get a very expressive synthesizer at the expense of producing twice as much data per second as the sampled audio. However, other synthesizers can provide a more compact representation, resulting in a smaller model which will be faster to train and run. The authors are currently working on implementing both an AM and a 2-operator FM differentiable synthesizer. These simple synthesizers will provide us a better understanding of the capabilities and nuances of a differentiable synth, its performance, how to integrate them in the existing toolchain, and how to modify the model architecture to fit different synthesizers.

Preprocessing of f0: Even if the model is able to transfer the timbre perfectly, following the "Garbage in, garbage out" concept, the quality of the output will be affected by the quality of the latent vectors. If the pitch estimation is not accurate, the resulting audio will present pitch artifacts. A quick solution can be to extract f0 from MIDI. While the resulting f0 is going to be precise, it is going to lack expressiveness. Solutions as the one proposed in [9] can add expressiveness to the MIDI data.

Explore the creative possibilities of the model: The creative possibilities offered by the DDSP architecture are immense, either with low fidelity, glitchy results as explored in this work, or with more realistic results by applying additional conditioning. Some of the possibilities are pitch- and time-shifting, lyric translation, voice morphing, change of singing style (e.g., to and from opera, pop, blues), tremolo and vibrato removal or addition, to name just a few. Working with clean data and synthesizing singing in different (or fictional) languages would be also interesting for the future.

Acknowledgments

The authors would like to thank the following singers for their generous collaboration providing the audio files: Servando Carballar from Aviador Dro (`https://www.instagram.com/el_aviador_dro/`), Belén Chanes from Lkan (`https://www.instagram.com/lkan_oficial/`) and Eva Páez (`http://evapaez.es/`).

7. REFERENCES

[1] J. Sundberg, "The singing voice," in *The Oxford handbook of voice perception*, S. Frühholz and P. Belin, Eds. Oxford University Press, 2018, ch. 6.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] X. Serra and J. Smith, "Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

[4] P. R. Cook, "Singing voice synthesis: History, current work, and future directions," *Computer Music J.*, vol. 2, no. 4, 1996.

[5] J. Engel, L. H. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=B1x1ma4tDr

[6] E. Gómez, M. Blaauw, J. Bonada, P. Chandna, and H. Cuesta, "Deep learning for singing processing: Achievements, challenges and impact on singers and listeners," *arXiv preprint arXiv:1807.03046*, 2018.

[7] G.-M. Hutter, "Timbre transfer on singing voices," Master's thesis, ETH Zurich, 2020.

[8] G. Fabbro, V. Golkov, T. Kemp, and D. Cremers, "Speech synthesis and control using differentiable DSP," *arXiv preprint arXiv:2010.15084*, 2020.

[9] N. Jonason, B. Sturm, and C. Thomé, "The control-synthesis approach for making expressive and controllable neural music synthesizers," in *2020 AI Music Creativity Conference*, 2020.

[10] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "Crepe: A convolutional representation for pitch estimation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 161–165.

[11] J. Engel, R. Swavely, L. H. Hantrakul, A. Roberts, and C. Hawthorne, "Self-supervised pitch detection with inverse audio synthesis," in *International Conference on Machine Learning, Self-supervised Audio and Speech Workshop*, 2020. [Online]. Available: https://openreview.net/forum?id=RlVTYWhsky7

[12] J. I. Godino-Llorente, P. Gomez-Vilda, and M. Blanco-Velasco, "Dimensionality reduction of a pathological voice quality assessment system based on gaussian mixture models and short-term cepstral parameters," *IEEE transactions on biomedical engineering*, vol. 53, no. 10, pp. 1943–1953, 2006.

EXPLORING MODALITY-AGNOSTIC REPRESENTATIONS FOR MUSIC CLASSIFICATION

Ho-Hsiang WU[1], **Magdalena FUENTES**[1,2], and **Juan P. BELLO**[1,2]

[1]*Music and Audio Research Laboratory*, **New York University**, New York, NY USA
[2]*Center for Urban Science and Progress*, **New York University**, New York, NY USA

ABSTRACT

Music information is often conveyed or recorded across multiple data modalities including but not limited to audio, images, text and scores. However, music information retrieval research has almost exclusively focused on single modality recognition, requiring development of separate models for each modality. Some multi-modal works require multiple coexisting modalities given to the model as inputs, constraining the use of these models to the few cases where data from all modalities are available. To the best of our knowledge, no existing model has the ability to take inputs from varying modalities, e.g. images or sounds, and classify them into unified music categories. We explore the use of cross-modal retrieval as a pretext task to learn modality-agnostic representations, which can then be used as inputs to classifiers that are independent of modality. We select instrument classification as an example task for our study as both visual and audio components provide relevant semantic information. We train music instrument classifiers that can take both images or sounds as input, and perform comparably to sound-only or image-only classifiers. Furthermore, we explore the case when there is limited labeled data for a given modality, and the impact in performance by using labeled data from other modalities. We are able to achieve almost 70% of best performing system in a zero-shot setting. We provide a detailed analysis of experimental results to understand the potential and limitations of the approach, and discuss future steps towards modality-agnostic classifiers.

1. INTRODUCTION

Musical objects and concepts appear in different heterogeneous data modalities, including but not limited to audio, images, text and scores, where sonic, visual and tactile modalities contribute to the overall experience. However, most music information retrieval (MIR) research has largely focused on developing systems that interact with a single modality, requiring development of separate models for audio, image or text, and over simplifying the musical modeling. There are approaches that exploit multiple modalities [1–4], but existing multi-modal systems in the context of MIR require *coexisting* modalities as inputs [5–10], which is a big constrain for their deployment since it limits the scope of systems to only work when the modality they have been design for is at hand.

In a context of rapidly increasing availability of information in all forms (video, audio, text, etc) it is desirable that models are able to overcome this single-modality limitation and can interact with information in any common form, for instance, a system able to classify musical instruments by the way they look and sound. To the best of our knowledge, no existing model in the context of MIR can be used if one of those modalities is missing (e.g. if it was trained with audio and text, can not be used in a dataset with only audio).

Based on recent work [11] we hypothesize that *modality-agnostic* systems can be developed by learning joint representations from different modalities when they represent the same concepts. If the embedding of an image of a guitar and the sound of a guitar are similar to each other (i.e. grouped closely in the embedding space) but different from those of a piano, we can build classification systems that would work with either image or audio. This would allow to train models in settings where there is big amounts of data from one modality but not from the other, but still be able to work in both cases.

This type of approach, often called *translation* since it implies "translating" one modality to another (e.g. being able to retrieve an image with a description of it) has received renewed attention recently given the combined efforts of the computer vision and natural language processing communities, and has been gaining more interests in the MIR community [12–18]. Recently, it has been proposed to learn translated representations using self-supervision [11] which is very promising since it doesn't rely on human-annotated data, but has the drawback of requiring millions pairs of raw data to train embedding models from scratch. We propose an intermediate solution, to use pre-trained embeddings and only learn the translation between them in a self-supervised manner, as a way of relaxing the amount of computation time and data needed for training the system.

In this paper, we take the first steps towards modality-agnostic music classification. We focus on the problem of classifying musical instruments using audio and/or image. We investigate the use of pre-trained audio and image embeddings in combination with training translation models to obtain a joint representation, in a self-supervised setting. We use the learned representations to train modality-

agnostic classifiers in a supervised manner, and we investigate the performance of the classifier compared to its single-modality counterpart in different scenarios, including one with varying amount of data available from either modality. Our implementation is available in `https://github.com/hohsiangwu/crossmodal`.

2. METHOD

Our method is summarized in Figure 1. It consists of three different stages: 1) First, we select a set of pre-trained embeddings from both audio and image, and translate or project the pre-trained embeddings into a common space, either by training a translation model, or simply using principal component analysis (PCA) to convert both embeddings to the same dimension; 2) We then conduct a study to find the best combination by comparing configuration performances in cross-modal retrieval; and 3) We use the resulting joint embeddings to train a classifier in a supervised setting and study the performance of the different configurations (i.e. translation vs. PCA vs. single modality) with different amount of data from each modality. We explain the different stages of the method in the following.

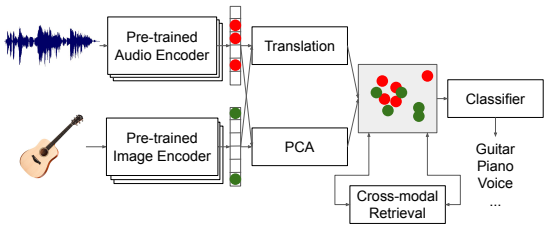


Figure 1. Method overview. The pre-trained audio and image embeddings are projected to a joint space by either the translation model or PCA, and the obtained embeddings are used to train the classifier in the downstream task: musical instrument classification. Cross-modal retrieval is used to select best configuration of pre-trained embeddings.

2.1 Pre-trained Image and Audio Embeddings

We select a set of state-of-the-art embeddings for both image and audio. For image embeddings, we use two pre-trained embedding models provided by *keras* library [1], trained using the ImageNet [19] dataset on a classification task. In particular, we use VGG Net [20] and ResNet [21]. These are both deep convolutional neural networks based architectures. We refer to [20, 21] for further details. For both models, we remove the last layer and apply average pooling to get the final image embeddings.

We also use pre-trained models to obtain the audio embeddings, particularly VGGish [22] , and YamNet. Finally, we use the open source implementation of OpenL3 [2] [4] trained with music data from AudioSet [23] to obtain another pair of image and audio embeddings.

[1] `https://keras.io/api/applications/`
[2] `https://github.com/marl/openl3`

Embedding model	# Parameters	Output dimension
OpenL3 (Image)	4.7M	8192
VGG16	15M	512
ResNet50	23.6M	2048
OpenL3 (Audio)	9M	6144
VGGish	62M	128
YamNet	3.2M	1024

Table 1. Overview of pre-trained image and audio embedding models.

In Table 1 we summarize the characteristics of each image and audio embedding model. Pre-trained VGG and ResNet image embeddings, VGGish and YamNet audio embeddings are trained on classification tasks, while OpenL3 is trained with audio-visual correspondence without labeled data.

To select the best combination, we evaluate how good the different pairs of embeddings blend together in a common space using a translation model. To quantify the success of this translation, we perform cross-modal retrieval (i.e. retrieve the image of an instrument using it's respective sound and vice versa) as further explained in Section 3.3. Our reasoning behind this is that for the modality-agnostic classifier to be successful, the embeddings should be very close to each other in the joint embedding space, and so they should be accurate in a cross-modal retrieval task when retrieving examples by distance. We select the best performing pair of audio and image embeddings and use it for the following stages.

2.2 Translation and Dimensional Reduction

We explore two ways of relating the audio and image embeddings: translation and a simple dimensional matching with PCA. For translation, in the self-supervised learning literature, various metric learning losses are used to learn a shared embedding space [24–26]. In particular, the *Contrastive loss* [27, 28] works well empirically with a careful selection of negative samples. It aims to minimize the distance of a given sample to positive examples (i.e. samples semantically related) while increasing the distance to negative examples concurrently. For the translation layer, we implement a 2 layer multi-layer perceptron (MLP) network, with pre-trained image and audio embeddings as inputs and train using contrastive loss, with cosine distance. We train the translation model using sample pairs from both modalities *without labels*, in particular we use the Musical Instruments AudioSet subset, as explained in Section 3. The output dimension is 128 for all of our embeddings.

As baseline, we apply PCA to each pre-trained embedding model to reduce their dimension to 128, and we train a single-modality classifier as well as a multi-modality classifier with such embeddings. The idea is to understand whether a simple solution is enough to build a modality-agnostic classification system, where the classifier is mainly responsible for the work of translating the modalities and learning the mapping to the labels.

2.3 Classification

We work with random forest classifiers. We train multi-modal (MM) classifiers either with translation (MMT) or PCA (MMP) and single-modality (SM) classifiers using dimension-reduced embeddings from audio (SMA) or image (SMI). We study how translation affects performance in scenarios with different amount of data used to train the classifiers. We do so by training the MMT and MMP with data from one modality and testing in the other, which we call *target modality*. We incorporate data from the target modality to the training of the classifiers by batches and see the impact in performance.

3. EXPERIMENTAL DESIGN

3.1 Dataset

Subset	Stage	# Samples
Translation	Train translation	130k
Cross-modal	Evaluate translation	10k
Classification	Train classifier	16.2k
	Evaluate classifier	1.8k

Table 2. Overview of subsets used for training and evaluation.

We use non-overlapping subsets of AudioSet [23] for the cross-modal experiments, the training of the translation model and the classifier. AudioSet is a multi-modal dataset containing YouTube videos with weak audio labels for a diverse set of real-world situations. We follow [11] and [4] by getting samples labeled at least with a descendant of "Musical instrument", "Singing" and "Tools". We then carefully split the dataset into three subsets, one for evaluating the pre-trained embedding combinations and select the best pair, another for training the translation model, and the last one for the downstream musical instrument classification task. From all qualified videos, we sample 1 second audio and one video frame as image within the second period. The assumption is that image and audio from roughly the same timestamp contain highly related semantic content. For evaluating the cross-modal retrieval experiments, we use a total 10k image/audio pairs. We call this subset the *cross-modal-subset*. We use 130k pairs to train the translation model, which is roughly half the amount of data used to train end-to-end models in [11]. We call this the *translation-subset* as shown in Table 2.

For the classification task, we carefully curated samples from 18 classes. Our categories include mapping from "Violin, fiddle" to "violin", "Choir" to "voice" and both "Drum" and "Drum kit" to "drums", and the remaining are "accordion", "banjo", "cello", "clarinet", "flute", "guitar", "mandolin", "organ", "piano", "saxophone", "synthesizer", "trombone", "trumpet", "ukulele", "cymbals". We manually audited the quality of the test set removing irrelevant samples (e.g. those labeled by piano but with image of an album cover with no piano on it) until we had 1,000 samples per instrument. Having a balanced dataset for the training of the classifier is important to prevent issues at this stage interfering with the assessment of the embeddings performance. We formulate the classification problem as a multi-class problem, where samples labeled only once from the above categories are selected. The result classification-subset consists of a balanced dataset with 16200 training samples and 1800 testing samples (10% split). We call it the *classification-subset*.

3.2 Model implementation

For ResNet and VGG image embeddings, we use the *preprocess* function provided from keras to normalize the pixel values. And we follow the pre- and post-processing steps of VGGish and YamNet [3]. For training the translation model, we do not use labels. Instead we randomly sample batches of size 4096 from translation-subset, extract pre-trained embeddings from both modalities, train a 2 layer MLP with both input dimensions as original pre-trained embeddings, 256 middle dimension, and 128 output dimension, implemented with PyTorch [4]. We use pairs of both modalities sampled from the same clip as positive examples, and other samples in the same batch as negative examples, with margin value as 1.0. We optimize using Adam optimizer with learning rate as 0.001, and we apply early stopping criteria on validation loss with patience as 5 epochs. For cross-modal retrieval, we take the outputs of translation model with corresponding pre-trained embeddings of the 10k image/audio pairs from cross-modal-subset, and use all embeddings from one modality as queries to fetch top 30 closest embeddings from another modality. For training the classification model, we use a random forest classifier from scikit-learn [5], with maximum depth set to 32, and 100 estimators.

3.3 Evaluation metrics

For the evaluation of the cross-modal retrieval results we follow the setup from [11]. We use *normalized discounted cumulative gain* (NDCG) score considering 30 elements. This score is a measure of ranking quality between 0 and 1 (from low to high quality), which assesses the gain of an element based on a relevance score and its position in the result list. Following [11], we use the relevance $r = C - d$, where d is the distance in the taxonomy graph between two labels in the AudioSet ontology, $C = 21$ being the maximum distance. As the AudioSet ontology is defined, the top labels (e.g. "Music", "Musical instrument", "Tools", "Singing") are included in computing the relevance, which make most of the example relevant since most of them convey one of those labels. [6] Therefore, we removed those top labels while computing the NDCG. We report the results of audio-to-image and image-to-audio retrieval.

For the evaluation of the classification results we use the macro F-measure or F1 score. Finally to assess the structural properties of the embedding spaces generated by the translation or the PCA projections we compute

[3] https://github.com/tensorflow/models/tree/master/research/audioset

[4] https://pytorch.org/

[5] https://scikit-learn.org/

[6] See https://research.google.com/audioset/ontology/index.html for further details.

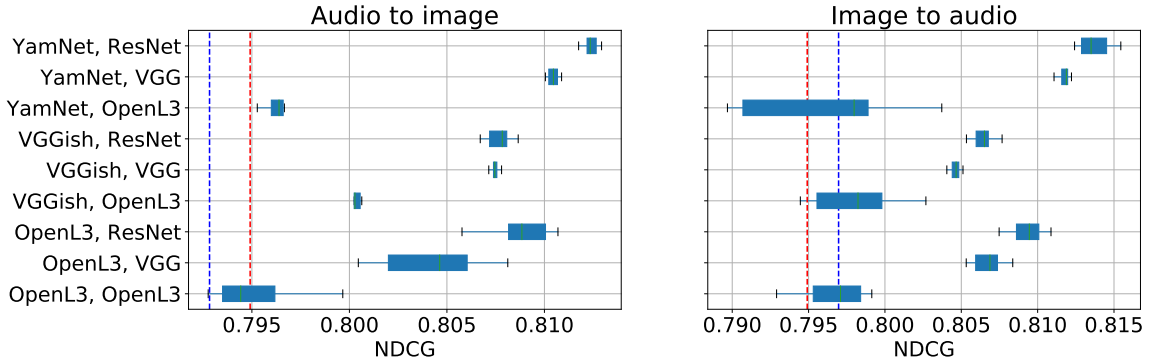


Figure 2. Cross-modal retrieval results with NDCG scores on the x-axis. On the y-axis we have different combinations of *audio, image* pre-trained embeddings used to train the translation model. The red dotted line is a random baseline, and the blue dotted line is OpenL3 (512 dimensions) for both image and audio without translation. We show cross-modal retrieval results of audio to image (left), and image to audio (right).

inter-cluster distances from the clusters of the different instrument classes and modality pairs before classification. For that, we take the test split of the classification-subset, compute average (centroids) of all the projected embeddings (PCA or translated) with the same modality and same instrument labels, and then compute pair-wise distance among modality/instrument clusters as the inter-cluster distance.

4. RESULTS AND DISCUSSIONS

4.1 What combination of pre-trained embeddings?

In this experiment we would like to determine which is the best audio-visual embeddings combination. We will have to simultaneously answer whether we are able to learn meaningful joint-embeddings from this data using translation, or if translation of pre-trained embeddings does not work at all.

To do so, we take all combination of audio and image embeddings and train the translation model with them, obtaining a total of nine separated translation models, i.e. nine mappings to joint embedding spaces. We then evaluate them using cross-modal retrieval in the cross-modal-subset explained in Section 3.1. The NDCG scores of different configurations are depicted in Figure 2, where both audio to image, and image to audio are shown. Following the ideas in [11], we use two baselines: random (red dotted line in Figure 2) which means randomly ordering the embeddings and get the first 30, and the OpenL3 (blue dotted line) image and audio embeddings both with 512 dimension [7], used for retrieval directly without translation.

The first observation is that the relative difference in NDCG scores between the baselines and our best performing model are comparable to those shown in [11], which is promising because it means that the translation model is effectively learning to relate the embeddings. Also unlike the systems in [11] which were trained from scratch, we obtained our joint embedding by translating pre-trained embeddings, and obtained competitive results. The random baseline performs better than OpenL3 for audio to image retrieval, which is consistent with the results reported in [11].

We observe a big gap in performance in all combinations that include the OpenL3 image embedding, which can be partially explained by the fact that VGG and ResNet greatly outperform that embedding in image classification downstream tasks, and thus more expressive embeddings would be better candidates for translation.

Overall, the combination of YamNet and ResNet performs the best across all configurations. We checked that is the case for the classification performance as well, therefore, we discuss only YamNet and ResNet results in the rest of the experiments.

4.2 How does translation affect performance?

We want to understand how translation affects the performance of a classifier in comparison to its multi-modal non-translated and single-modality counterparts. For that we compare their performance in the classification task, by training the classifier using embeddings from one modality and testing with embeddings from the other, and by adding batches of the training modality with balanced number of instrument classes by bits. We use the classification-subset for this experiment. The results of this process are shown in Figure 3, where the macro F1 scores are reported for a test set of only images (left) and only audio (right).

No data from target modality. First, we discuss the results in the 0 point of the x-axis, corresponding to the performance of the classifiers without any data from the target modality (e.g. when testing in image, only training the MM classifiers with audio). We see a similar and expected behaviour in both modalities: the MMP classifier is guessing some classes right (very little), probably exploiting some unintended relations between YamNet and ResNet embeddings after PCA, and the MMT classifier clearly outperforms the others, being able to achieve al-

[7] Note that the OpenL3 implementation allows for multiple output dimensions, and we choose 512 here for both embeddings to be comparable.

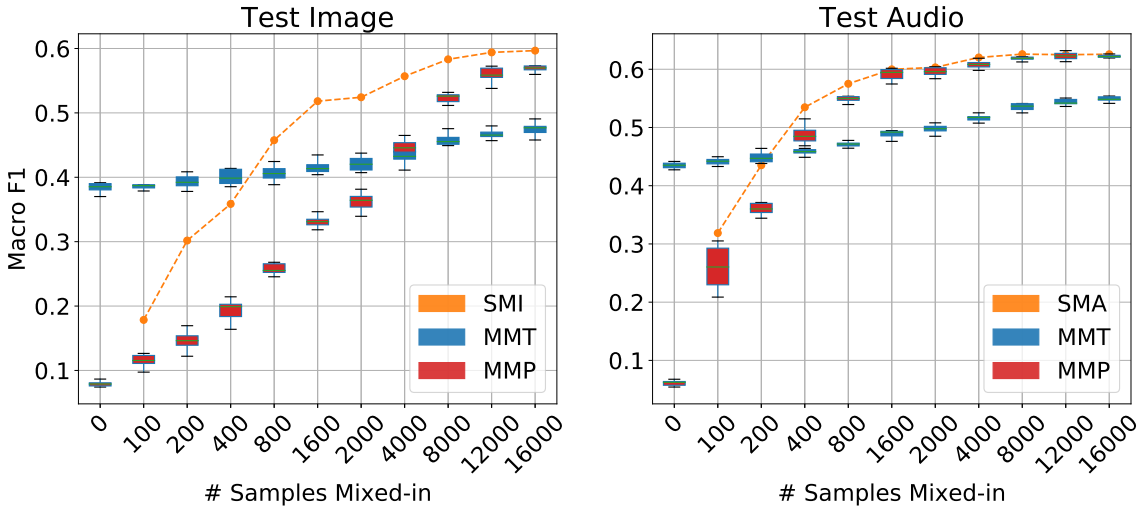


Figure 3. Instrument classification results in different modalities: image (left) and audio (right). The orange dashed line is a SM classifier trained and tested with same modality, image (I) or audio (A). Blue (translated) and red (PCA) boxes are MM classifiers trained with data from different modality to the test set, with the x-axis indicating the number of samples from the target modality mixed-in in during training (e.g. when testing on image, MMT and MMP are trained with audio, and image data is mixed in by bits).

most 70% of the best performance already. This confirms what we saw from the cross-modal retrieval examples, that the translation is doing a meaningful mapping, and further this allows to learn from one modality and test in another in a zero-shot fashion.

Adding data from target modality. However, for an ideal translation, image and audio embeddings would be interchangeable. That means that the performance of MMT without seeing any embedding from the target modality or after seeing all of them should be the same (since no *new* data would be added to the classifier). And so, the blue curve we see in Figure 3 with a small slope should flat at the maximum performance independently of the test data we add in. The fact that the performance of MMT increases by adding this data is showing that the translation failed in combining *some* meaningful information. And this makes the MMT classifier's performance to fall behind when all the data from both modalities are available (point 16000 in the x-axis of Figure 3).

Observing the performance of MMP, we also see that with the right amount of data and without translation, the classifier is able to learn the mapping between embeddings and classify the instruments correctly. This is an interesting result since it implies that for specific tasks with available labeled data from both (or multiple) modalities, it is enough to train a classifier to learn to deal with different modalities all together and be able to work with whatever modality is available at inference time with almost the same performance than a classifier fully dedicated to one modality.

4.3 What is translation doing?

We want to understand what is happening during translation that the MMT classifier is struggling to keep up with the others when enough data from both modalities is available, and why it does not reach best performance starting from the zero-shot setting. We compare the structure of the non-translated and translated embeddings before feeding them into the classifier. In particular we measure the distance between the cluster centroids of the different clusters of classes in each setting. Figure 4 shows the pairwise distance for the different modalities and instrument classes. On the left we see the non-translated embeddings sorted by modality, and on the right we see translated embeddings sorted by instrument class. The two figures show that the embedding spaces are indeed different, and that the translation is structuring and bringing together the audio and image embeddings of the same class (shown as small 2x2 squares on the diagonal), i.e. grouping the embeddings by concepts. This makes sense and explains why the translation works in the zero-shot setting, and is interesting considering that the translation layer is trained in an unsupervised manner.

However, there are noticeable small distance square blocks in both images: the one in the left on the top left, shows that the audio embeddings are closer to each other, which is an artifact of YamNet embeddings. This is not what happens with the ResNet embeddings, which after the PCA projection are sometimes closer to other image embeddings but also sometimes closer to audio embeddings. An exception to this are the image embeddings for voice, cymbals and synthesizer, which are very different from all the other embeddings.

The other big block where embeddings are grouped to-

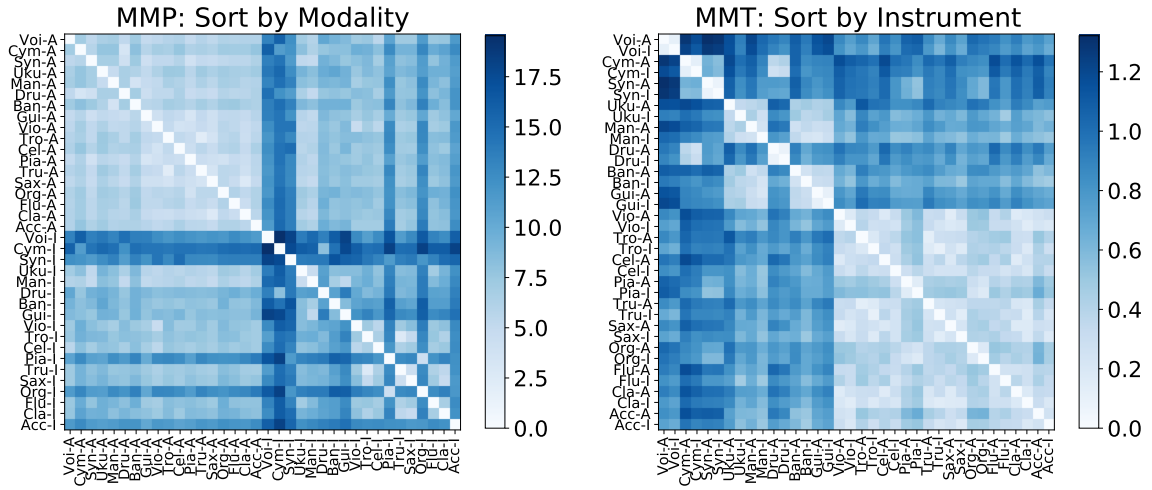


Figure 4. Pair-wise distance of inter-cluster centroid for modality and instrument classes. On the left we have PCA sorted first by modality. On the right we have translated sorted first by instrument class. Note: This is results of classification-subset but before training the classifier.

gether, in the right image of Figure 4, shows that the translation is bringing together some classes (e.g. accordian, clarinet, flute, organ, saxophone, etc.) that should not be blend together, and the overall distances in the translated embedding space are smaller than in the non-translated one. Observing the class distribution of the data used for training the translation model in Figure 6 (note the labels were not use in the training, only here for the analysis), we observe it is skewed and the classes with fewer number of instances correlate with most of the confused ones in the translated embeddings. We think that this is probably causing the classification performance of MMT to drop with respect to MMP and SM. The exception is voice, which we speculate has to do with an effect from the ResNet embedding which is very different from all other embeddings and we suspect that helped the translated cluster to be sufficiently different as well.

To see the correlation between our observations in the embedding space and the classification performance, we look at the per instrument F1 and confusion matrices of the MMT classifier, using all data from both modalities as shown in Figure 5. In each figure we show the per instrument F1 on the left, and the confusion matrix of MMT on the right. Looking at the per instrument F1 on the left, we can also observe the correlations between less performant instrument classes with smaller distances in Figure 4 and number of fewer samples in Figure 6. Looking at the confusion matrix of MMT on the right, we see that there are common mistakes of cymbal vs drum (both modalities), trombone vs trumpet (both modalities), guitar vs mandolin (mostly image), and organ vs piano (mostly image), which make sense because of the acoustic or visual similarity of those instruments in each modality. We observe a similar trend in the confusions made by the MMP classifier, but in a lesser extent (which explains the better performance).

To sum up, we believe that the bias of label distribution

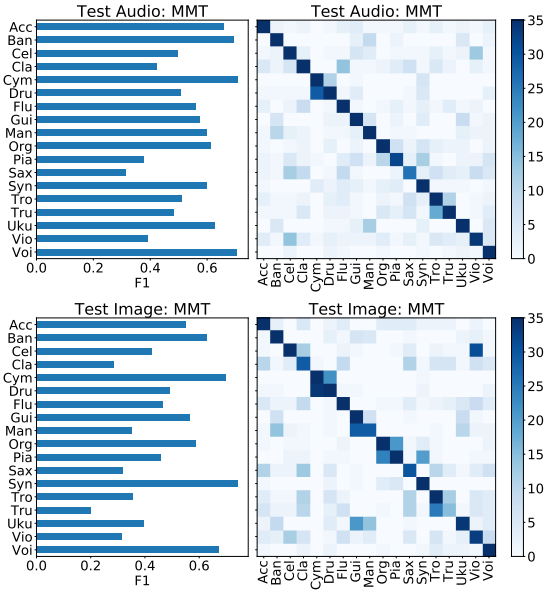


Figure 5. Classification results: training with all data using both modalities and testing in audio (top) and image (bottom). Per instrument F1 on the left, confusion matrix on the right.

in the data we used for training translation is the main cause of the performance drop in classification, and this is a trade-off of self-supervised learning without using the labels. We plan to explore in the future unsupervised methods for sample selection to balance the training set used for the translation model, such as determinant point processes [29].

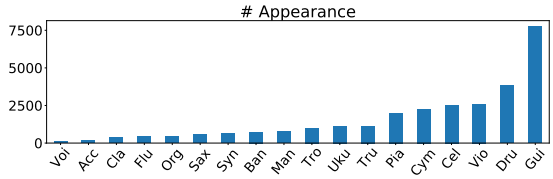


Figure 6. Number of samples containing classification instrument labels in translation-subset. We show here only the classes under study in the classification task.

5. CONCLUSIONS AND FUTURE WORK

In this work we propose and investigate modality-agnostic representations for music classification. We first present a study on different combinations of pre-trained audio and image embeddings to determine the best configuration to obtain modality-agnostic representations via cross-modal evaluation. We then use this representation to train instrument classifiers, comparing with non-translated and single-modality baselines. We show promising results as well as interesting potential applications using data from one modality to train and another modality to test with reasonable performance (almost 70% of best performing system in a zero-shot setting). We also investigate how biases in the training data used for the translation affect the classification performance.

For future work, we are interested in exploring sampling methods [30, 31] that could help balance the training set to obtain a more unbiased translation model, which from our analysis could lead to better performance. Also, we are interested in exploring the joint training of the translation and classification models, instead of the sequential method proposed in this paper. Furthermore, we think that exploring novel loss functions specifically for multi-modal data will also be an interesting direction as most of the current contrastive methods are applied to single modality. This work presented first steps and analysis towards the use of modality-agnostic representations in music, which we consider to be a promising idea in the context of MIR since it allows the use of data from different datasets and modalities in a flexible way, relaxing concerns about data scarcity and other data-availability related issues.

Acknowledgments

This work is partially supported by the National Science Foundation award #1544753. Magdalena Fuentes is a faculty fellow in the NYU Provost's Postdoctoral Fellowship Program at the NYU Center for Urban Science and Progress and Music and Audio Research Laboratory.

6. REFERENCES

[1] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 2, pp. 423–443, 2018.

[2] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.

[3] H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. C. Platt *et al.*, "From captions to visual concepts and back," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1473–1482.

[4] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3852–3856.

[5] A. Yazdani, K. Kappeler, and T. Ebrahimi, "Affective content analysis of music video clips," in *Proceedings of the 1st international ACM workshop on Music information retrieval with user-centered and multimodal strategies*, 2011, pp. 7–12.

[6] A. Schindler and A. Rauber, "An audio-visual approach to music genre classification through affective color features," in *European Conference on Information Retrieval*. Springer, 2015, pp. 61–67.

[7] O. Slizovskaia, E. Gómez, and G. Haro, "Musical instrument recognition in user-generated videos using a multimodal convolutional neural network architecture," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, 2017, pp. 226–232.

[8] S. Oramas, F. Barbieri, O. Nieto, and X. Serra, "Multimodal deep learning for music genre classification," *Transactions of the International Society for Music Information Retrieval. 2018; 1 (1): 4-21.*, 2018.

[9] Z. Duan, S. Essid, C. C. Liem, G. Richard, and G. Sharma, "Audiovisual analysis of music performances: Overview of an emerging field," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 63–73, 2018.

[10] J. Choi, J. Lee, J. Park, and J. Nam, "Zero-shot learning for audio-based music classification and tagging," in *The 20th International Society for Music Information Retrieval Conference (ISMIR)*. International Society for Music Information Retrieval Conference (ISMIR), 2019, pp. 67–74.

[11] R. Arandjelovic and A. Zisserman, "Objects that sound," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 435–451.

[12] M. Dorfer, J. Hajič Jr, A. Arzt, H. Frostel, and G. Widmer, "Learning audio–sheet music correspondences for cross-modal retrieval and piece identification," *Transactions of the International Society for Music Information Retrieval*, vol. 1, no. 1, 2018.

[13] Y. Zhang, B. Pardo, and Z. Duan, "Siamese style convolutional neural networks for sound search by vocal imitation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 2, pp. 429–441, 2018.

[14] K. Lee and J. Nam, "Learning a joint embedding space of monophonic and mixed music signals for singing voice," in *The 20th International Society for Music Information Retrieval Conference (ISMIR)*. International Society for Music Information Retrieval Conference (ISMIR), 2019, pp. 295–302.

[15] B. Li and A. Kumar, "Query by video: Cross-modal music retrieval." in *ISMIR*, 2019, pp. 604–611.

[16] K. Watanabe and M. Goto, "Query-by-blending: A music exploration system blending latent vector representations of lyric word, song audio, and artist." in *ISMIR*, 2019, pp. 144–151.

[17] F. Zalkow and M. Müller, "Learning low-dimensional embeddings of audio shingles for cross-version retrieval of classical music," *Applied Sciences*, vol. 10, no. 1, p. 19, 2020.

[18] R. Arandjelovic and A. Zisserman, "Look, listen and learn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 609–617.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[22] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017, pp. 131–135.

[23] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[24] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[25] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 1735–1742.

[26] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep metric learning with angular loss," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2593–2601.

[27] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 297–304.

[28] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[29] A. Kulesza and B. Taskar, "Determinantal point processes for machine learning," *Machine Learning*, vol. 5, no. 2-3, pp. 123–286, 2012.

[30] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl, "Sampling matters in deep embedding learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2840–2848.

[31] M. Won, S. Oramas, O. Nieto, F. Gouyon, and X. Serra, "Multimodal metric learning for tag-based music retrieval," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 591–595.

ATTENTION-BASED PREDOMINANT INSTRUMENTS RECOGNITION IN POLYPHONIC MUSIC

Lekshmi C. REGHUNATH[1] and **Rajeev RAJAN**[1]

[1]*College of Engineering Trivandrum*, **APJ Abdul Kalam Technological University**, Trivandrum, Kerala, India

ABSTRACT

Predominant instrument recognition in polyphonic music is addressed using the score-level fusion of two visual representations, namely, Mel-spectrogram and modgdgram. Modgdgram, a visual representation is obtained by stacking modified group delay functions of consecutive frames successively. Convolutional neural networks (CNN) with an attention mechanism, learn the distinctive local characteristics and classify the instrument to the group where it belongs. The proposed system is systematically evaluated using the IRMAS dataset with eleven classes. We train the network using fixed-length single-labeled audio excerpts and estimate the predominant instruments from variable-length audio recordings. A wave generative adversarial network (WaveGAN) architecture is also employed to generate audio files for data augmentation. The proposed system reports a micro and macro F1 score of 0.65 and 0.60, respectively, which is 20.37% and 27.66% higher than those obtained by the state-of-the-art Han model. The experiments demonstrate the potential of CNN with attention mechanism on Mel-spectro/modgdgram fusion framework for the task of predominant instrument recognition.

1. INTRODUCTION

Predominant instrument recognition refers to the problem where the prominent instrument is identified from a mixture of instruments being played together [1]. The auditory scene produced by a musical composition can be regarded as a multi-source environment, where different sound sources are played at various pitches and loudness, and even the spatial position of a given sound source may vary with respect to time [2]. In polyphonic music, the interference of simultaneously occurring sounds makes instrument recognition harder. Automatic identification of lead instrument is important since it helps to enhance fundamental music information retrieval (MIR) tasks like source separation [2] auto-tagging [3], and automatic music transcription [4].

Han *et al.* [1] employed the Mel-spectrogram-CNN approach for predominant instrument recognition in polyphonic music using an aggregation strategy over sliding windows. Pons *et al.* [5] analyzed the architecture of Han *et al.* in order to formulate an efficient design strategy to capture the relevant information about timbre. Both approaches were trained and validated by the IRMAS dataset of polyphonic music excerpts. Detecting the activity of music instruments using a deep neural network (DNN) through a temporal max-pooling aggregation is addressed in [6]. The paper [7] employed an attention mechanism and multiple-instance learning (MIL) framework to address the challenge of weakly labeled instrument recognition in the OpenMIC dataset. Dongyan Yu *et al.* [8] employed a network with an auxiliary classification scheme to learn the instrument categories through multitasking learning. Gomez *et al.* [9] investigated the role of two source separation algorithms as pre-processing steps to improve the performance in the context of predominant instrument detection tasks. It was found that both source separation and transfer learning could significantly improve the recognition performance, especially for a small dataset composed of highly similar musical instruments. In [10], the Hilbert-Huang transform (HHT) is employed to map one-dimensional audio data into two-dimensional matrix format, followed by CNN to learn the affluent and effective features for the task. In [11] an ensemble of VGG-like CNN classifiers, trained on non-augmented, pitch-synchronized, tempo-synchronized, and genre-similar excerpts of IRMAS for the proposed task. The modified group delay feature (MODGDF) has already been proposed for pitched musical instrument recognition in an isolated environment [12] and polyphonic predominant instrument recognition [13]. Bosch *et al.* improved the algorithm proposed in [2] with source separation in a pre-processing step [14]. While the commonly applied mel frequency cepstral coefficients (MFCC) feature is capable of modeling the resonances introduced by the filter of the instrument body, it neglects the spectral characteristics of the vibrating source, which also play its role in human perception of musical sounds and genre classification [15]. Incorporating phase information is an effective attempt to preserve this neglected component. It has already been established in the literature that the modified group delay function emphasizes peaks in spectra well [16]. The idea of including modgdgram, GAN-based data augmentation strategy, and CNN with multi-head attention are the main contributions of the proposed scheme.

The rest of the paper is organized as follows. Section 2 gives an overview of the proposed model. The model architecture is described in Section 3. The performance evaluation is described in Section 4, followed by results in

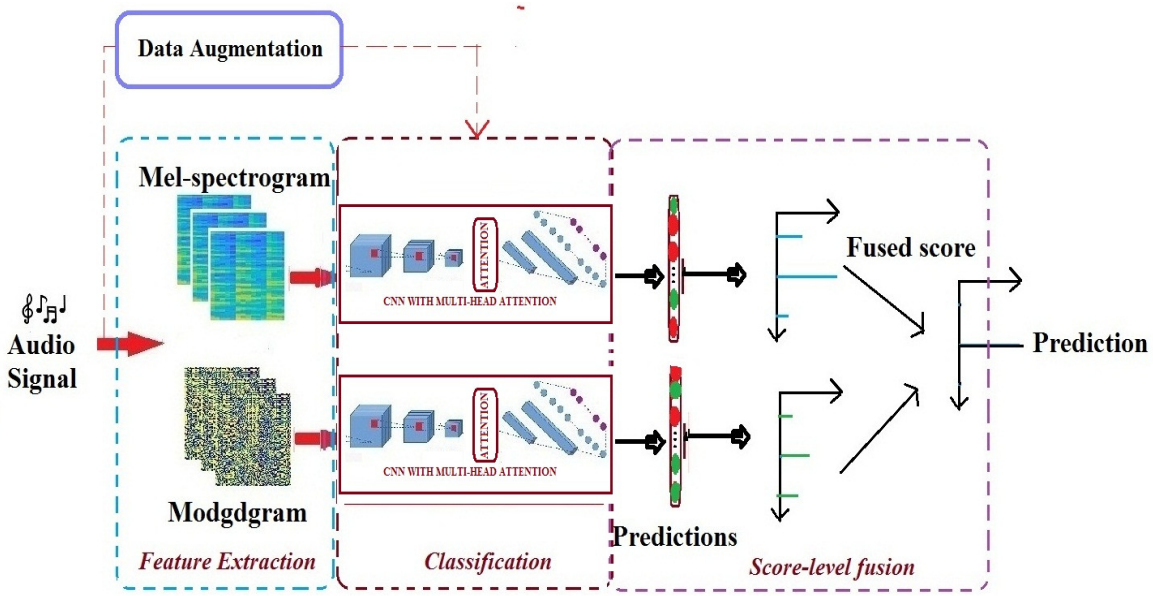


Figure 1. Block diagram of the proposed method of predominant instrument recognition.

Section 5. The paper is concluded in Section 6.

2. SYSTEM DESCRIPTION

The block diagram of the proposed method of predominant instrument recognition is illustrated in Figure 1. In the proposed model, CNN with multi-head attention is used to learn the distinctive characteristics from Mel-spectro/modgd-gram to identify the leading instrument in polyphonic context. As a part of data augmentation, additional training files are generated using WaveGAN. During the testing phase, the probability value at the output nodes of the trained model is treated as the score corresponding to the input test file. The input audio file is classified to the node which gives the maximum score during testing. In the fusion framework, the individual scores of Mel-spectro/modgd-gram experiments are fused at the score-level to make a decision. The fusion score S_f, is obtained by,

$$S_f = \beta S_{spectro} + (1-\beta) S_{modgd} \quad (1)$$

where $S_{spectro}$, S_{modgd}, β are the Mel-spectrogram score, modgdgram score and weighting constant, respectively. β = 0.5 is chosen empirically in our experiment. The performance of the proposed system is compared with that of Han's model and a DNN framework. Feature extraction is described in the following subsections.

2.1 Mel-spectrogram

Mel-spectrogram is widely used in recent music processing applications [17, 18]. Mel-spectrogram approximates how the human auditory system works and can be seen as the spectrogram smoothed, with high precision in the low frequencies and low precision in the high frequencies [19]. It is computed with a frame size of 50 ms and a hop size of 10 ms with 128 bins for the given task.

2.2 Modified group delay functions and Modgdgram

Group delay features are being employed in numerous speech and music processing applications [16, 20–22]. The group delay function is defined as the negative derivative of the unwrapped Fourier transform phase with respect to frequency. Modified group delay functions (MODGD), $\tau_m(e^{j\omega})$ are obtained by,

$$\tau_m(e^{j\omega}) = \left(\frac{\tau_c(e^{j\omega})}{|\tau_c(e^{j\omega})|}\right)(|\tau_c(e^{j\omega})|)^a, \quad (2)$$

where,

$$\tau_c(e^{j\omega}) = \frac{X_R(e^{j\omega})Y_R(e^{j\omega}) + Y_I(e^{j\omega})X_I(e^{j\omega})}{|S(e^{j\omega})|^{2b}}. \quad (3)$$

The subscripts R and I denote the real and imaginary parts, respectively. $X(e^{j\omega})$, $Y(e^{j\omega})$ and $S(e^{j\omega})$ are the Fourier transforms of signal, $x[n]$, n.$x[n]$ (signal multiplied with index), and the cepstrally smoothed version of $X(e^{j\omega})$, respectively. a and b ($0 < a,\ b \leq 1$) are introduced to control the dynamic range of MODGD [16]. Modgdgram is the visual representation of MODGD with time and frequency in the horizontal and vertical axis, respectively. The amplitude of the group delay function at a particular time is represented by the intensity or color in the third dimension. Modgdgrams are computed with a frame size of 50 ms and hop size of 10 ms using a and b values of 0.9 and 0.5 respectively.

3. MODEL ARCHITECTURE

3.1 Fusion without attention

First, we designed a three-layer CNN to encode the Mel-spectrogram or modgdgram images. Filters of sizes 32, 64, and 192 are used in the convolutional layers. Each convolutional layer is followed by 2x2 max-pooling. ReLU is used as the activation in the hidden layer and the same padding is employed to maintain the spatial resolution. We used filters with a very small 2×2 receptive field, with a fixed stride size of 1. For implementing fusion without attention, we flattened the last convolutional layer followed by fully connected layers. 20% of training data is used for tuning the hyperparameters during training. Softmax is used as the activation function for the output layer with 11 outputs. The number of parameters learned by the baseline model is 1830315 parameters.

3.2 Attention model

Multi-head attention is employed after three convolutional layers. It expands the model's ability to focus on different positions of Mel-spectrogram/modgdgram. For constructing the attention the principle behind is to create smaller linear representations of the same block by splitting the content of a block into query vectors(q), key vectors(k), and value vectors(v). Using key and query vectors we can create weights for the value vectors that will be used to create the output vector. In the paper [23], such attention blocks are used to encode and decode the sentences. The various parameters chosen are shown in Table 1.

N	l	d	h	dv	$dout$	dk
6	6*6 = 36	64*3 = 192	8	8*3 = 24	32	36

Table 1. Various hyperparameters (top row) and its values (bottom row) selected for attention model.

where N represents the number of encoders or decoders present in the self-attention layer, l is the number of blocks in the feature map the convolutional network made, d is the dimension of block, dv is the dimension of linear space the input to be projected, $dout$ is the output of the block after applying attention and h is the number of heads or number of projections for each block. dk is the dimension of the query vector. $q1, k1, v1$ are the query, key and value vectors of input and $q2, k2, v2$ are the h projections with size dv of the $q1, k1, v1$ vectors. Then each query vector is multiplied with each key vector to get the softmax predictions over h value vectors. The outputs from all 8 attention heads are concatenated to form a single output vector before passing it through the feed-forward network. After the attention layer, a normalization layer is also added to increase the speed of convergence. It makes the tensor have a standard normal distribution, at the same time it acts as another smaller attention by deleting some dimensions of the vector that are not important. The norm layer is followed by flattened layers. The network is trained using adam optimizer with a learning rate of 0.001. The network learns the model with 473163 parameters which are approximately 4 times smaller than the baseline model without attention. The model summary of the proposed method of CNN with multi-head attention is shown in Table 2.

Input size	Description
1x28x28	Modgdgram /Mel-spectrogram
32x28x28	2x2 Convolution, 32filters
32x14x14	2x2 Max-pooling
64x13x13	2x2 Convolution, 64 filters
64x7x7	2x2 Max-pooling
192x6x6	2x2 Convolution, 192 filters
192x36	Reshape
32x36	Multi-head attention
32x6x6	Reshape
32x6x6	Normalization layer
1152	Flattened and fully connected
256	Dense
11	Softmax

Table 2. Proposed CNN architecture with multi-head attention.

4. PERFORMANCE EVALUATION

4.1 Dataset

IRMAS dataset [2], comprising eleven classes, is used for the evaluation. The classes include cello (Cel), clarinet (Cla), flute (Flu), acoustic guitar (Gac), electric guitar (Gel), organ (Org), piano (Pia), saxophone (Sax), trumpet (Tru), violin (Vio) and human singing voice (Voice). The training data consists of 6705 audio files with excerpts of 3 s from more than 2000 distinct recordings. Since the dataset consists of testing audio samples with multiple predominant instruments as labels, we have considered all the audio files with a single predominant instrument (single label) during the testing phase.

4.2 Data augmentation using WaveGAN

GAN has been successfully applied to a variety of problems in image generation [24] and style transfer [25]. WaveGAN v2 is used here to generate polyphonic files with the leading instrument required for training. WaveGAN is similar to DCGAN, which is used for Mel-spectrogram generation, in various music processing applications. The transposed convolution operation of DCGAN is modified to widen its receptive field in WaveGAN. Specifically, longer one-dimensional filters of length 25 are used instead of two-dimensional filters of size 5x5 and are upsampled by a factor of 4 instead of 2 at each layer. The discriminator is also modified similarly, using length 25 filters in one dimension [26]. The output dimensionality of WaveGAN v2 is 65536 samples (corresponding to 4.01 s of audio at 16 kHz). For the generator, the input is a random noise uniformly distributed between -1 and 1. For training, the WaveGAN optimizes WGAN-GP using Adam for both generator and discriminator. A constant learning rate of

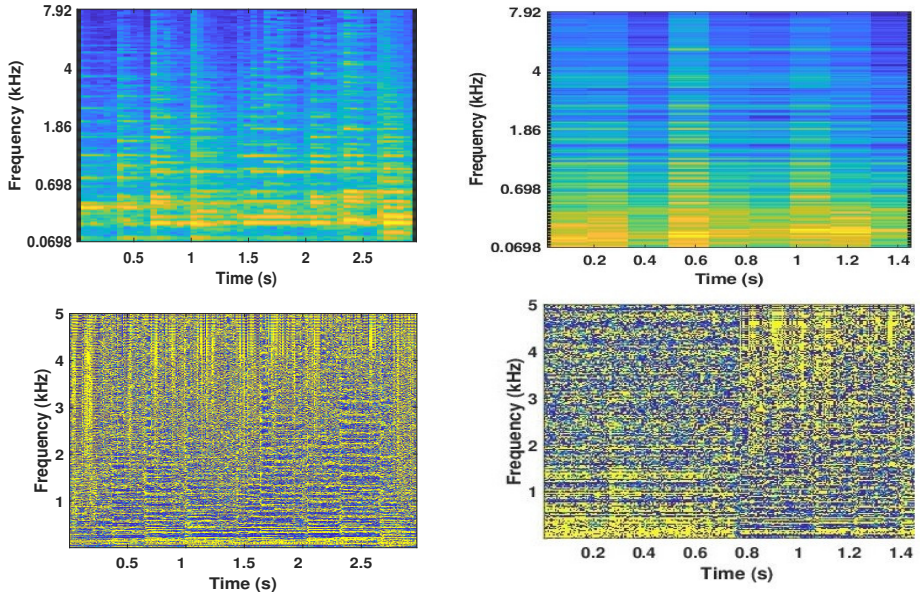


Figure 2. Visual representation of an audio excerpt with acoustic guitar as leading, Mel-spectrogram of original and WaveGAN-generated (Upper pane left and right). Modgdgram of original and WaveGAN-generated (Lower pane left and right).

0.0001 is used with $\beta_1 = 0.5$ and $\beta_2 = 0.9$. WaveGAN is trained for 2000 epochs on the three sec audio files of each class to generate similar audio files based on a similarity metric (s) [27] with a criteria $s > 0.1$. A total of 6585 audio files with cello (625), clarinet (482), flute (433), acoustic guitar (594), electric guitar (732), organ (657), piano (698), saxophone (597), trumpet (521), violin (526) and voice (720) are generated.

The quality of generated files is evaluated using a perception test. It is conducted with ten listeners to assess the quality of generated files for 275 files covering all classes. Listeners are asked to grade the quality by choosing one among the five opinion grades varying from poor to excellent quality (scores, 1 to 5). A mean opinion score of 3.64 is obtained. This value is comparable to the mos score obtained in [26] and [28] using WaveGAN. The generated files are denoted by $Train_g$ and training files available in the corpus are denoted by $Train_d$. Mel-spectrogram and modgdgram of natural and generated audio files for acoustic guitar are shown in Figure 2. The experiment details and a few audio files can be accessed at https://sites.google.com/view/audiosamples-2020/home/instrument

4.3 Experimental set-up

The experiment is progressed in three phases namely Mel-spectrogram-based, modgdgram-based, and score-level fusion-based. 1305 polyphonic files comprising eleven classes with a single label are used for the testing phase. The performance of the proposed method is compared with that of Han's model [1]. As different from our approach, they used a sliding window to perform short-time analysis, and sigmoid outputs were aggregated by taking class-wise average. After normalization, the candidate with maximum probability is assumed to be the most predominant instrument. Han's baseline model is implemented for the given experiment with 1 s slice length for performance comparison [1]

A DNN framework on musical texture features (MTF) is also experimented with to examine the performance of deep learning methodology on handcrafted features. MTF includes MFCC (13 dim), spectral centroid, spectral bandwidth, root mean square energy, spectral roll-off, and chroma STFT. The features are computed with a frame size of 40 ms and a hop size of 10 ms using Librosa framework [2]. DNN consists of seven layers, with increasing units from 8 to 512. ReLU has been chosen for hidden layers and softmax for the output layer. The network is trained for 500 epochs using Adam optimizer with a learning rate of 0.001.

Since the number of annotations for each class was not equal, we computed precision, recall, and F1 measures for both the micro and the macro averages. For the micro averages, we calculated the metrics globally, thus giving more weight to the instrument with a higher number of appearances. On the other hand, we calculated the metrics for each label and found their unweighted average for the macro averages. Overall accuracy is also used as a metric for performance evaluation. ‘

5. RESULTS AND ANALYSIS

The overall performance of different phases of the experiment is tabulated in Table 3. Fusion with Attention (Fusion-Attn) network achieved micro and macro F1 mea-

[1] https://github.com/Veleslavia/EUSIPCO2017
[2] https://librosa.org/doc/latest/tutorial.html

SL.No	Class	MTF-DNN			Han's Model			Mel-spectrogram-Attn			Modgdgram-Attn			Fusion-Attn		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	Cello	0.54	0.59	0.56	0.55	0.44	0.49	0.48	0.70	0.57	0.09	0.15	0.12	0.79	0.67	**0.73**
2	Clarinet	0.15	0.40	0.22	0.23	0.64	0.33	0.74	0.68	0.71	0.10	0.28	0.15	0.54	0.80	**0.65**
3	Flute	0.17	0.21	0.19	0.54	0.54	0.54	0.66	0.60	0.63	0.21	0.16	0.18	0.68	0.72	**0.70**
4	Acoustic guitar	0.59	0.39	0.47	0.62	0.51	0.56	0.61	0.44	0.51	0.61	0.38	0.47	0.66	0.57	**0.61**
5	Electric guitar	0.56	0.46	0.51	0.57	0.51	0.54	0.53	0.64	0.58	0.49	0.47	0.48	0.66	0.68	**0.67**
6	Organ	0.22	0.45	0.29	0.20	0.42	0.27	0.26	0.69	0.38	0.12	0.18	0.14	0.30	0.62	**0.40**
7	Piano	0.70	0.36	0.47	0.72	0.58	0.64	0.68	0.66	0.67	0.64	0.55	0.59	0.73	0.71	**0.72**
8	Saxophone	0.03	0.40	0.06	0.13	0.60	0.21	0.10	0.70	0.18	0.03	0.20	0.05	0.18	0.70	**0.29**
9	Trumpet	0.14	0.57	0.23	0.30	0.86	0.44	0.86	0.43	0.57	0.19	0.36	0.24	0.59	0.71	**0.65**
10	Violin	0.24	0.53	0.33	0.43	0.58	0.49	0.85	0.31	0.45	0.38	0.45	0.42	0.56	0.51	**0.53**
11	Voice	0.55	0.34	0.43	0.69	0.55	0.61	0.74	0.47	0.57	0.68	0.54	0.60	0.78	0.61	**0.68**
	Macro	0.35	0.43	**0.34**	0.45	0.57	**0.47**	0.59	0.57	**0.53**	0.32	0.34	**0.31**	0.59	0.66	**0.60**
	Micro	0.39	0.39	**0.39**	0.54	0.54	**0.54**	0.56	0.56	**0.56**	0.43	0.43	**0.43**	0.65	0.65	**0.65**

Table 3. Precision (P), recall (R), and F1 score for the experiments with data augmentation.

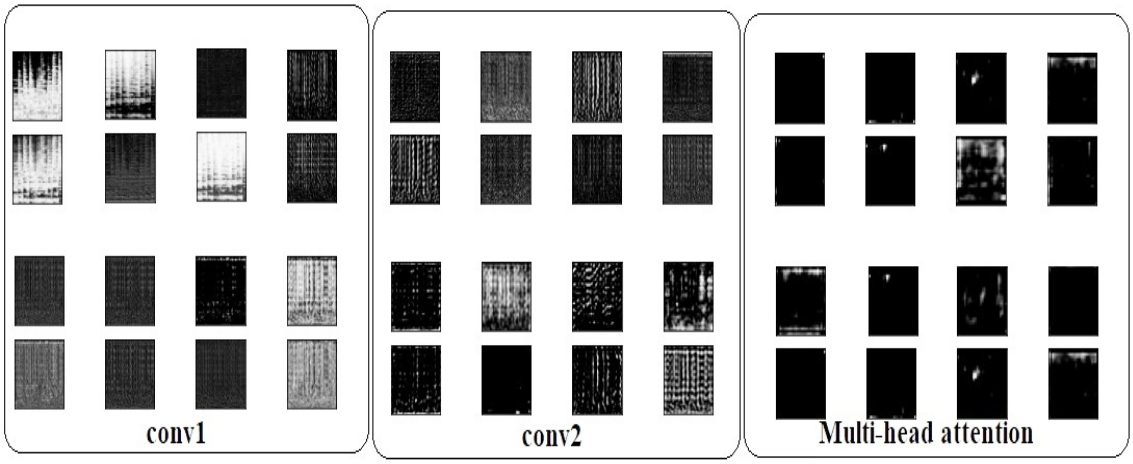


Figure 3. Visualisation of feature maps of convolutional layers and attention. The upper pane represents the feature maps for Mel-spectrogram inputs and the lower pane represents the feature maps for modgdgram inputs.

sures of 0.65 and 0.60, respectively. The State-of-the-art Han model reports micro F1 and macro F1 scores of 0.54 and 0.47, respectively. Micro F1 and macro F1 are 20.37% and 27.66% higher than those obtained for the baseline model. Modgdgram added complementary information to the spectrogram approach. Conventionally, the spectrum-related features used in instrument recognition take into account merely the magnitude information. However, there is often additional information concealed in the phase, which could be beneficial for recognition [12]. Han's model and the proposed Mel-spectrogram approach show similar performance with better performance for the proposed architectural choice. It is worth noting that modgdgram itself outperforms the MTF-DNN methodology. It reveals the importance of phase information in musical processing tasks.

5.1 Effect of attention

Attention mechanisms have become an integral part of compelling sequence modeling and transduction models since these models show superior quality while being more parallelizable and requiring significantly less time to train. [23]. It is applied to a variety of speech and music processing applications like speech emotion recognition [29], music instrument recognition [7], music generation [30] etc. For polytimbral music instrument recognition attention model focus on specific time segments in the audio relevant to each instrument label. The ability of the attention model to weigh relevant and suppress irrelevant predictions for each instrument leads to better classification accuracy [7]. Compared to self-attention the multi-head attention gives the attention layer multiple representation subspaces, and as the image passes through different heads

Sl.No	Model	Micro			Macro		
		P	R	F1	P	R	F1
1	Fusion-without Attn ($Train_d + Train_g$)	0.54	0.54	0.54	0.46	0.58	0.48
2	Fusion-Attn ($Train_d$)	0.57	0.57	0.57	0.53	0.63	0.54
3	Fusion-Attn ($Train_d + Train_g$).	0.65	0.65	0.65	0.59	0.66	0.60

Table 4. Performance comparison of the models with and without data augmentation.

Sl.No	Model	Micro			Macro		
		P	R	F1	P	R	F1
1	Bosch et al. [14]	0.50	0.50	0.50	0.41	0.45	0.43
2	Han et al./1446k [1]	**0.65**	0.56	0.60	**0.54**	0.51	0.50
3	Single-layer/62k [5]	0.61	0.52	0.56	0.52	0.48	0.48
4	Multi-layer/743k [5]	0.65	0.54	0.59	0.55	0.52	0.52
5	**Fusion-Attn** ($Train_d + Train_g$)**/473k**	0.63	**0.63**	**0.63**	0.51	**0.55**	**0.52**

Table 5. Performance comparison for IRMAS dataset.

predictions about the predominant instruments are more refined than employing single head self-attention. Another important point is that it requires very few trainable parameters to learn the model, which helps to reach convergence faster than the models employing CNN alone. The significance of attention in the proposed model can be analyzed from Table 4. Fusion without Attention reports micro and macro F1 scores of 0.54 and 0.48 respectively. Fusion with Attention reports micro and macro F1 scores of 0.65 and 0.60, respectively, with an improvement of 20.37% and 25% higher than that obtained by Fusion without Attention.

Visualization of the feature maps extracted from the first two convolutional layers and attention layer is shown in Figure 3. It is created with 8 feature maps as subplots. The feature maps close to the input detect small or fine-grained detail, whereas attention layer feature maps capture more general and refined features for predominant instrument recognition.

5.2 Effect of data augmentation

For deep learning, the number of training examples is critical for the performance compared to the case of using hand-crafted features because it aims to learn a feature from the low-level input data [1]. The significance of data augmentation in the proposed model can be analyzed from Table 4. Fusion-Attn without data augmentation ($Train_d$) reports micro and macro F1 score of 0.57 and 0.54 respectively. Fusion-Attn ($Train_d + Train_g$) reports micro and macro F1 score of 0.65 and 0.60, respectively, with improvement of 14.03% and 11.11% higher than that obtained by Fusion ($Train_d$).

5.3 Multiple predominant instrument recognition

The IRMAS dataset contains testing files of variable length and has multiple predominant instruments. For our initial work, we considered only the variable-length polyphonic testing files with a single predominant instrument. The same experiment is repeated for multiple predominant instrument recognition using the entire 2874 testing files available in the corpus. For that, we trained our networks using fixed-length excerpts containing a single predominant instrument and estimated an arbitrary number of instruments from variable-length audio files having multiple predominant instruments.

The standard metrics for various algorithms on the IRMAS corpus are reported in Table 5. The number of trainable parameters is also indicated. Bosch *et al.* [14] algorithm used typical hand-made timbral audio features with their frame-wise mean and variance statistics to train SVMs with source separation technique called flexible audio source separation framework (FASST) in a preprocessing step. The state-of-the-art Han model [1] reports micro and macro F1 score of 0.60 and 0.50 respectively. Han *et al.* [1] developed a deep CNN for instrument recognition based on Mel-spectrogram inputs. Pons *et al.* [5] customized the architecture of Han *et al.* and introduced two models, namely, single-layer and multi-layer approaches. They used the same aggregation strategy as that of Han's model by averaging the softmax predictions and finding the candidates with a threshold of 0.2. As different from the existing approaches, we estimated the predominant instrument using the entire Mel-spectrogram without sliding or aggregation analysis. As our Mel-spectrogram/modgdgram inputs pass through multiple heads the presence of predominant instruments is refined from the simultaneously occurring partials. Our Fusion approach reports a micro and macro F1 score of 0.63 and 0.52, which is a 5 % and 4 % increase from Han's model. Also, our proposed method shows better micro and macro recall than the existing techniques. Our proposed method reports a micro and macro recall of 0.63 and 0.55, which is an 12.5 % and 7.84 % increase from Han's model. It reveals the importance of the attention

mechanism in predicting multiple instruments. A significant improvement is also observed over the method proposed in [14]. Also, the fusion network reports better results with very less trainable parameters, compared to existing techniques. In [7], the usage of an attention layer was shown to improve classification results in the OpenMIC dataset, when applied to a set of Mel-spectrogram features extracted from a pre-trained VGG net. While the work [7] focusses on Mel-spectrogram, we experimented with the effect of phase information along with magnitude information. The experimental results in the paper show the potential of Mel-spectrogram and modgdgram on recognizing predominant instruments in a polyphonic environment with multi-head attention.

6. CONCLUSIONS

We presented an Attention-based predominant instrument recognition system using Mel-spectro/modgd-gram inputs. CNN with multi-head attention is used to capture the instrument-specific characteristics and then do further classification. The proposed method is evaluated using the IRMAS dataset. Data augmentation is also performed using WaveGAN. The fusion framework outperforms the latest model proposed by Han *et al.* The results show the potential of score-level fusion of magnitude and phase-based approaches and the attention mechanism empowers the network to focus on specific regions of Mel-spectrogram/modgdgram in predominant instrument recognition in polyphonic music.

Acknowledgments

The first author would like to acknowledge the CERD of APJ Abdul Kalam Technological University, Trivandrum, Kerala, India for providing a Ph.D. fellowship.

7. REFERENCES

[1] Y. Han, J. Kim, and K. Lee, "Deep convolutional neural networks for predominant instrument recognition in polyphonic music." *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 1, pp. 208–221, 2017.

[2] F. Fuhrmann and P. Herrera, "Polyphonic instrument recognition for exploring semantic similarities in music," *in Proc. of 13th International Conference on Digital Audio Effects DAFx10, Graz, Austria*, vol. 14, no. 1, pp. 1–8, 2010.

[3] J.-Y. Liu and Y.-H. Yang, "Event localization in music auto-tagging," *in Proc. of the 24th ACM international conference on Multimedia*, pp. 1048–1057, 2016.

[4] Z. Duan, J. Han, and B. Pardo, "Multi-pitch streaming of harmonic sound mixtures," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 1, pp. 138–150, 2013.

[5] J. Pons, O. Slizovskaia, R. Gong, E. Gómez, and X. Serra, "Timbre analysis of music audio signals with convolutional neural networks," *in Proc. of 25th European Signal Processing Conference (EUSIPCO)*, pp. 2744–2748, 2017.

[6] S. Gururani, C. Summers, and A. Lerch, "Instrument activity detection in polyphonic music using deep neural networks," *in Proc. of International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[7] S. Gururani, M. Sharma, and A. Lerch, "An attention mechanism for musical instrument recognition," *in Proc. of 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[8] D. Yu, H. Duan, J. Fang, and B. Zeng, "Predominant instrument recognition based on deep neural network with auxiliary classification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 852–861, 2020.

[9] J. S. Gómez, J. Abeßer, and E. Cano, "Jazz solo instrument classification with convolutional neural networks, source separation, and transfer learning." *in Proc. of International Society for Music Information Retrieval Conference (ISMIR)*, pp. 577–584, 2018.

[10] X. Li, K. Wang, J. Soraghan, and J. Ren, "Fusion of hilbert-huang transform and deep convolutional neural network for predominant musical instruments recognition," *in Proc. of 9th International conference on Artificial Intelligence in Music, Sound, Art and Design*, 2020.

[11] A. Kratimenos, K. Avramidis, C. Garoufis, A. Zlatintsi, and P. Maragos, "Augmentation methods on monophonic audio for instrument classification in polyphonic music," *in Proc. of 28th European Signal Processing Conference (EUSIPCO)*, pp. 156–160, 2021.

[12] A. Diment, P. Rajan, T. Heittola, and T. Virtanen, "Modified group delay feature for musical instrument recognition," *in Proc. of 10th International Symposium on Computer Music Multidisciplinary Reserach, Marseille, France*, pp. 431–438, May 2013.

[13] R. Ajayakumar and R. Rajan, "Predominant instrument recognition in polyphonic music using gmm-dnn framework," *in Proc. of International Conference on Signal Processing and Communications (SPCOM)*, pp. 1–5, 2020.

[14] J. J. Bosch, J. Janer, F. Fuhrmann, and P. Herrera, "Acomparison of sound segregation techniques for predominant instrument recognition in musical audio signals." *in Proc. of 13th International Society for Music Information Retrieval Conference (ISMIR)*, 2012.

[15] F. Fuhrmann., "Automatic musical instrument recognition from polyphonic music audio signals," *PhD thesis, Universitat Pompeu Fabra*, 2012.

[16] H. A. Murthy and B. Yegnanarayana, "Group delay functions and its application to speech processing," *Sadhana*, vol. 36, no. 5, pp. 745–782, 2011.

[17] M. Sukhavasi and S. Adappa, "Music theme recognition using cnn and self-attention," *arXiv:1911.07041*, 2019.

[18] D. Ghosal and M. H. Kolekar, "Music genre recognition using deep neural networks and transfer learning." *in Proc. of Interspeech*, pp. 2087–2091, 2018.

[19] D. O'shaughnessy, "Speech communication: human and machine," *Universities press*, pp. 1–5, 1987.

[20] R. Rajan and H. A. Murthy, "Two-pitch tracking in co-channel speech using modified group delay functions," *Speech Communication*, vol. 89, pp. 37–46, 2017.

[21] ——, "Music genre classification by fusion of modified group delay and melodic features," *in Proc. of National Conference on Communications*, 2017.

[22] ——, "Melodic pitch extraction from music signals using modified group delay functions," *in Proc. National Conference on of the Communications (NCC)*, pp. 1–5, February 2013.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *in Proc. of Neural Information Processing Systems (NIPS)*, 2017.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Aaron Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[25] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," *in Proc.of International Conference on Machine Learning*, pp. 1857–1865, 2017.

[26] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," *in Proc. of International Conference on Learning Representations (ICLR)*, pp. 1–16, 2019.

[27] A. Madhu and S. Kumaraswamy, "Data augmentation using generative adversarial network for environmental sound classification," *in Proc, of 27th European Signal Processing Conference (EUSIPCO)*, pp. 1–5, 2019.

[28] G. Atkar and P. Jayaraju, "Speech synthesis using generative adversarial network for improving readability of hindi words to recuperate from dyslexia," *Neural Computing and Applications*, pp. 1–10, 2021.

[29] Y. Yu and Y.-J. Kim, "Attention-lstm-attention model for speech emotion recognition and analysis of iemocap database," *Electronics*, vol. 9, no. 5, p. 713, 2020.

[30] G. Keerti, A. Vaishnavi, P. Mukherjee, A. S. Vidya, G. S. Sreenithya, and D. Nayab, "Attentional networks for music generation," *arXiv preprint arXiv:2002.03854*, 2020.

RUFFLE: A USER-CONTROLLABLE MUSIC SHUFFLING ALGORITHM

Giorgio PRESTI (giorgio.presti@unimi.it) (0000-0001-7643-9915)[1],
Federico AVANZINI (federico.avanzini@unimi.it) (0000-0002-1257-5878)[1],
Adriano BARATÈ (adriano.barate@unimi.it) (0000-0001-8435-8373)[1],
Luca Andrea LUDOVICO (luca.ludovico@unimi.it) (0000-0002-8251-2231)[1], and
Davide Andrea MAURO (maurod@marshall.edu) (0000-0001-8437-4517)[2]

[1] *Laboratorio di Informatica Musicale (LIM), Department of Computer Science*, **Università degli Studi di Milano**, Italy
[2] *Department of Computer and Information Technology*, **Marshall University**, USA

ABSTRACT

Music shuffling is a common feature, available in most audio players and music streaming platforms. The goal of this function is to let songs be played in random, or constrained random, order. The results obtained by in-use shuffling algorithms can be unsatisfactory due to several factors including: the variability of user expectations to what constitutes a "successful" playlist, the common bias of being unable to recognize true randomness, and the tendency of humans to find nonexistent patterns in random structures. In this paper, a new shuffling algorithm called *Ruffle* is presented. *Ruffle* lets the user decide which aspects of the music library have to be actually shuffled, and which features should remain unchanged between consecutive extractions. First, an online survey was conducted to collect users' feedback about the characteristics used for shuffling. It is worth noting that, in general, the algorithm could address any metadata and/or audio extracted feature. Then, in order to test the algorithm on personal playlists, a Web version based on Spotify API has been released. For this reason, a second survey is marking an ongoing effort placed on validating the effectiveness of the algorithm by collecting users' feedback, and measuring the level of user satisfaction.

1. INTRODUCTION

The shuffle feature of a music player is a well known function that lets the user listen to each song of the personal catalog in a pseudo-random order. In [1] the shuffle mode is defined as "a control that paradoxically involves a renunciation of control on the part of the user".

People may use such a function for many reasons, including the reduction of boredom in listening, the need to keep the attention alive, the search for "serendipity" [2], etc. For a general overview of shuffle in music, see [3, 4].

As shown in [5], humans are hardly capable not only of identifying, but also of generating random sequences of numbers. This is mainly due to two phenomena: i) the *gambler fallacy* [6], that consists in thinking that, in a memory-less random draw, previous drawings have an influence on the actual ones, and ii) the *clustering illusion* [7], that consists in erroneously considering small clusters of samples from random distributions to be non-random.

An ambiguity when talking about the randomness of a playlist in natural language is whether to compute the order of the songs with or without regard of their metadata. In the latter case, music pieces can be seen as completely different objects drawn at random, even if they may share some common values for metadata; in the former case, the expected behavior is to travel the song space with the longest possible path, such that the distance of each consecutive song is maximized with respect to some properties. This can be seen as an interesting variant of the Travelling Salesman Problem, which is, in turn, a special case of the Longest Path Problem.

In order to manage the complex and multifaceted problem of how to shuffle songs in a playlist, a new algorithm, called *Ruffle*, has been developed. *Ruffle* generates a pseudo-random sequence that matches the user's preferences about what must be shuffled, and to what extent. It does so by exploiting the gambler fallacy to present the problem to the users, which makes their choices, that in turn (together with what has been previously drawn) will affect the probability of the remaining songs in the list to be drawn for the next play.

The rest of the paper is organized as follows: Section 2 provides the state of the art about industry standards and commercial applications integrating a shuffle function; Section 3 presents a survey designed to understand user preferences about playlist shuffling, also discussing the collected results in order to guide the design of a new approach; Section 4 focuses on *Ruffle*, a user controllable shuffle algorithm; Section 5 provides an early assessment of *Ruffle*, presenting a web-based implementation that interfaces with Spotify API, and the outcomes of a survey where participants tested its functionalities in an actual scenario; finally, Section 6 draws the conclusions.

2. STATE OF THE ART

Before presenting the state of the art, it is worth remarking the distinction between a shuffle function and a music recommendation system. A shuffle algorithm simply re-

orders a given list of songs already available in the user's collection. This operation can also be performed by taking into account some preferences about the desired outcome, but no user data out of the scope of a listening session is involved. Conversely, a recommendation (or suggestion) system aims at proposing content that is not necessarily already in the user's collection [8]. Moreover, this task is typically accomplished by modeling user's preferences and making inferences about musical tastes based on long-term behaviors. Some algorithms may combine the two approaches, e.g. by changing the shuffle strategy based on user's actions such as song skips, implicitly interpreted as a manifestation of dislike towards a particular genre or artist. For the sake of clarity, this paper deals with the shuffle problem only, without making any inference.

A number of algorithmic approaches can be used to produce a randomly shuffled sequence of elements in a list.

The Fisher-Yates shuffle, whose original form dates back to 1938, represents a way for generating a random permutation of a finite sequence. The algorithm puts all the elements into an unordered set, i.e. a container that stores unique elements in no particular order, and establishes the next element in the sequence by randomly drawing one of them from the set until no elements remain. The algorithm produces an unbiased permutation, i.e. every permutation is equally likely. First described in [9] and originally conceived as a pencil-and-paper method, its computer implementation was documented in [10] and later published in [11]. A critical issue of the algorithm is that one of its steps requires to pick a random number, but only a high-quality unbiased random number source can guarantee unbiased results. The Fisher-Yates algorithm is at the base of the implementation proposed in [12].

The application domain represented by a music collection introduces some additional requirements. In fact, single music pieces often share some characteristics with others, and a good shuffle algorithm could try to maximize the distance considering also such similarities. For example, multiple songs in a dataset could be authored and/or performed by the same artist(s), could belong to the same genre, could share the same ensemble, etc.

This problem has been addressed in [13], where a form of *balanced shuffle* is proposed. The idea is to merge pre-shuffled playlists, each one made of pieces belonging to the same group, where a group contains elements sharing similar properties as it regards a given dimension. For the sake of clarity, let the chosen dimension be the genre: a group contains all classical pieces, another group all jazz pieces, and so on; each sub-playlist is shuffled; finally, a merge-and-mix operation is conducted to transform pre-shuffled sub-playlists into a single playlist.

Singh *et al.* [14] proposed a form of *predictive shuffling* that can provide automated dynamic-based shuffling according to the user's preferences by taking into account various parameters (e.g. genre, artist, play duration and release date) and selecting the next song accordingly.

An original approach described in the literature is so-called *responsive shuffling* [15], that benefits from temporal, spatial, and mental context awareness. Based on interactive soundscape concepts and wearable-computing technologies, this work proposes context-driven playlist shuffling for music listening in mobility.

Concerning documented shuffle functions in commercially-available services, Spotify [1] initially started from the Fisher-Yates algorithm, then evaluated the balanced shuffle described above, and finally moved to a method which is claimed to be inspired by image dithering [16].

Pandora [2] implements different forms of shuffle depending on user's privileges: premium subscribers have more flexibility with their shuffle options and can shuffle songs as well as the content on playlists or stations. The set of music features that can be considered comes from Pandora's *Music Genome Project*, an effort to "capture the essence of music at the most fundamental level" [17] employing over 450 attributes, called *genes*, to describe songs and a complex algorithm to organize them.

Although some of the shuffling algorithms mentioned above implement advanced features (such as considering a subset of metadata to vary or, conversely, to keep fixed among consecutive draws), such a mechanism is usually hidden from the user, with little to no room for controls, so the results may not match personal preferences.

3. TUNING TO LISTENERS' PREFERENCES

3.1 Survey design

In order to understand in detail users preferences and desiderata about the shuffle listening experience, an online survey was carried out.

The survey is made of three parts. The first part collects general information about the user such as the average daily listening time, the preferred player/platform, how often a shuffle function is used when organizing a playlist, and the level of satisfaction with the shuffle behavior.

The second part consists of an open question inviting users to reflect on which, according to their opinion, should be the aspects used in constraining a shuffle algorithm (e.g. preserving the same tempo, or genre), and what constitutes a "successful" playlist.

Finally, users are requested to express their preferences regarding specific parameters: 1. genre, 2. artist, 3. album, 4. BPM, 5. language, 6. publication date, 7. instrumental/-song, and 8. allowing repetitions.

Further comments are allowed as free text in the last page before submission. Surveys have been conducted in Italian and they are here translated to English.

3.2 Survey results and discussion

3.2.1 Users overview

In total, 84 answers were collected from Italian users. The subjects are 56% males, 42% females, and 2% not specified; 64% were 17-24 years old, 13% in the range 25-30, 15% in range 31-50, and 8% > 50.

Listening habits: 11% listens to less than 1 hour of music per day, 44% listens to 1-2 hours, 24% 2-3 h, and 21%

[1] https://www.spotify.com
[2] https://www.pandora.com

more than 3 h. Most used listening platforms are Spotify, Youtube, and personal libraries, in particular the preferred combinations were Spotify+Youtube 37%, Spotify only 23%, Youtube only 6%, and Spotify+Youtube+Personal 5%; the remaining 29% uses a variety of combination of the above services and others, such as, in preference order, Amazon Music, Apple Music, Google Play Music, Deezer, TIDAL, web or analog radio, Bandcamp.

About the use of the shuffle function, 29% rarely uses it, 18% sometimes, and 53% frequently. 24% are not satisfied by the shuffle function, 43% are neither satisfied nor unsatisfied, and 33% are satisfied.

Use of shuffle correlates weakly with age (spearman $R = 0.3$, $p < 0.005$), in particular almost all subjects with age > 50 rarely uses shuffle, while subjects with age < 50 present more heterogeneous behavior.

3.2.2 Open question about shuffling

Concerning the open question, the 84 participants provided a number of different comments. A manual clustering revealed three wide categories: [3]

- Randomness (21 comments);
- Music properties (48 comments);
- Suggestion systems (28 comments).

The comments regarding randomness may be summarized by the following 4 sentences:

R1 *All available songs should be shuffled as randomly as possible, without distinctions* (7 comments);

R2 *A song should not be re-played until all songs have been played* (5 comments);

R3 *Shuffle should produce very different sequences across listening sessions* (8 comments);

R4 *Shuffle outcome should be stored in case the listener is interested in navigating the playlist* (1 comment).

In regard to R1, Fisher-Yates should be the preferred choice, unfortunately none of the 7 users really meant it, since when asked whether a feature may be kept constant or variable across plays, or whether the feature must be irrelevant, almost all answered the artist and album must vary, the genre must stay constant, and they left only BPM, language, and publication year as actually random. This is no surprise, since it has already been discussed how humans, on average, have no precise insight on how randomness behaves.

Even if R2 may seem obvious, when explicitly asked about this aspect, not all subjects agreed (more on this in Sec. 3.2.3).

Note that R3 may seem to suggest something different from the equiprobability of the sequences that make Fisher-Yates an adequate algorithm, nevertheless, if we interpret this as an extension of the previous one (i.e. "A song should not be re-played until all songs have been played, *even across different sessions*"), then Fisher-Yates is still a valid approach.

The behavior described in R4 is usually implemented in modern platforms as "user history", or directly by design when the shuffle function works offline, and returns a playlist instead of one-by-one songs.

The comments regarding music properties may be summarized by the following 6 sentences:

M1 *Shuffle should be aware of music properties* (48 comments). In particular, the following were cited: Genre (19 mentions); Artist (10 mentions); Generic "properties" (9 mentions); BPM (4 mentions); Mood (3 mentions); Album (2 mentions); Key (2 mentions); Release year (1 mention); Song duration (1 mention); Instruments (1 mention); Timbre (1 mention).

M2 *All proposed songs should be strictly coherent in terms of one or more properties* (17 comments). In particular, it was specified in 3 comments that a user-selected song should set the baseline;

M3 *Some incoherence can be tolerated in order to avoid monotony* (4 comments);

M4 *Changes of music properties should be gradual* (8 comments);

M5 *Consecutive songs should be very different in terms of one or more properties* (4 comments);

M6 *It should be possible to choose between "coherent" and "incoherent" behavior* (2 comments).

Properties frequently cited in M1 confirm the validity of the items selected for the multiple choices version of the question (see Sec.3.2.3), except for the mood, which was not considered. The 9 mentions of some "generic property" may be implemented by adding audio features as well as metadata in the algorithm, nevertheless, not all players support this level of detail. It should be noted that some of these sentences are in contrast with each other (e.g. M2 and M5), thus a good shuffling feature should be tunable, as suggested in M6. Finally, M4 implies that it is worth considering the introduction of some sort of "memory" in the shuffle algorithm.

The comments regarding recommendation systems may be summarized by the following 6 sentences:

S1 *Priority should be given to frequently listened songs* (5 comments);

S2 *Some inference about musical taste is expected* (9 comments);

S3 *Some music discovery algorithm is expected* (4 comments);

S4 *Song skipping should be considered in future draw* (7 comments);

[3] Number of comments may add up to more than number of participants, since many participants provided multiple comments; moreover, also the expanded views of the clusters may add up to more than the total cluster comments count, since some comments are two-folded, e.g. comment "Artist and genre may stay constant" expresses an interest both in the "artist" and "genre" metadata, and the fact that the shuffle may produce coherent outcomes accordingly.

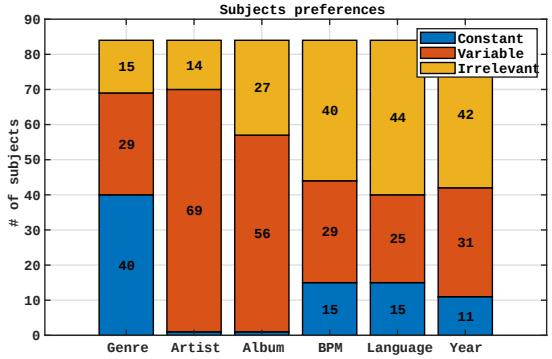


Figure 1: Hypothetical preferences expressed by subjects.

S5 *Context information (such as current time and geolocation) should be considered* (2 comments);

S6 *The algorithm should teach something to the user* (1 comment).

Sentences like S1-3 and S5 will be ignored, since they must rely on user information and recommendation algorithms.

The feature described in S4 is somehow borderline between shuffling and suggestion, it may be worth considering it in future works, and let this work focus on pure shuffling.

S6 is also related to a recommendation algorithm, nevertheless it provides a hint toward a shuffle algorithm that lets you pick a path in the songs feature space. Yet another feature that can be explored in the future.

3.2.3 Multiple-choice questions about shuffling

As a first question, subjects were asked to tell, for each metadata (genre, artist, album, BPM, language, and release year) if that should remain constant among plays, if it should vary, or if should be irrelevant. Figure 1 shows an overview of the answers. The sum of "constant" and "variable" preferences has been interpreted as a score of the wish to have control over it. All metadata received a score greater or equal to 50%, except for language, which is slightly less than 50%.

In more detail, 49 unique combinations of answers were given: 31 combinations were selected only once, 11 combinations appeared twice, and only the remaining 7 combinations were selected three or more times (these are visible in Table 1). This considerable variability, with some frequent choices, is two-folded: on one side the presence of frequent choices can suggest that some template behavior for shuffle algorithms may be useful, but on the other side, to meet the needs of most users, a fine tuning mechanism seems desirable.

To further investigate if a pattern is present when tolerating some small differences in the answers, answers to this question were clustered using linkage hierarchical clustering method. Specifically, the hamming distance was selected to compute distances between pairs of answers, and weighted average distance was used as linkage. The tree was cut so to retain most of the groupings of Table 1 without creating neither many small clusters nor large heterogeneous clusters. The resulting cluster's medioids are visible in Table 2.

Clustering revealed that most of the answers appearing only once can be considered very similar to the frequently given combinations, especially 1,2, and 7. The only new cluster (8) can be considered as a truly random version of 2, and one answer (with ID 9) resulted to be so different from the others to be considered the only true outlier.

In conclusion, 92% of the answers can be traced back to the 7 most populated clusters, and in particular clusters 1 and 2 captures almost half the total preferences.

Regarding the possibility to influence the shuffling algorithm by limiting the reproduction of vocal or instrumental tracks, 75% of users agreed.

On the possibility of repeating the same song twice in the same listening session, 79% of answers were negative, 15% and 5% of users considered it acceptable only after a long time or even after a short time, respectively, and the remaining 1% considered this feature to be irrelevant.

3.3 Lessons learnt

The survey highlighted a set of *desiderata* for the shuffling algorithm, which can be synthesized as follows:

D1 Songs should not be replayed until the whole song list has been played;

D2 Available properties such as metadata and audio features should be considered;

D3 A tuning mechanism between constant and variable behavior should be provided;

D4 The system is supposed to have a short-term memory to ease gradual changes in properties;

D5 There should be the ability to seed the algorithm by picking a reference song.

Answering D1 and D5 is trivial, while D2 and D3 are at the core of *Ruffle*. D4 has been implemented by introducing a *memory parameter* β which acts on the *responsiveness* of the system.

4. THE *RUFFLE* ALGORITHM

The rationale behind *Ruffle* is to update after each draw the probability of other songs to be drawn, according to songs properties and user settings. In particular, a similarity δ between the drawn song and the remaining ones is calculated for each considered property, and a probability weight is associated to the songs, which is either directly or inversely proportional to δ based on user decision.

4.1 Variables

Let $\boldsymbol{x}_N$ be a set of N songs:

$$\boldsymbol{x}_N = \{x_1, x_2, \cdots, x_n \mid 0 < n \leq N\}\,; \qquad (1)$$

ID	Genre	Artist	Album	BPM	Lang.	Year	Count	Mnemonic
1	V	V	V	V	V	V	8	Forced randomness
2	C	V	V	I	I	I	6	Genre exploration
3	I	I	I	I	I	I	4	True randomness
4	V	V	V	I	I	I	4	Enhanced randomness
5	C	V	V	V	C	I	3	Cultural niche
6	V	V	V	V	I	V	3	Tolerant randomness
7	V	V	V	C	I	C	3	Memorabilia DJ

Table 1: Most selected combinations. Some mnemonic names are given (V: Variable; C: Constant; I: Irrelevant).

ID	Genre	Artist	Album	BPM	Lang.	Year	Count	Mnemonic
1	V	V	V	V	V	V	19	Forced randomness
2	C	V	V	I	I	I	21	Genre exploration
3	I	I	I	I	I	I	7	True randomness
4	V	V	V	I	I	I	9	Enhanced randomness
5*	C	V	I	V	C	I	6	Refined cultural niche
6	V	V	V	V	I	V	5	Tolerant randomness
7	V	V	V	C	I	C	11	Memorabilia DJ
8	C	I	I	I	I	I	5	Genre strolling
9	C	I	V	C	I	V	1	Genre DJ

Table 2: Cluster's medioids. Some mnemonic names are given (V: Variable; C: Constant; I: Irrelevant).

let $\boldsymbol{c}_L$ be an execution queue of $L \leq N$ songs, initially empty:

$$\boldsymbol{c}_L = \{c_1, c_2, \cdots, c_l \mid 0 \leq l \leq L\}; \tag{2}$$

let $\boldsymbol{x'}_R$ be a subset of $\boldsymbol{x}_N$, composed of $R \leq N$ remaining songs, initialized as $\boldsymbol{x'}_R \equiv \boldsymbol{x}_N$:

$$\boldsymbol{x'}_R \subset \boldsymbol{x}_N; \tag{3}$$

let x_Δ be the last song inserted in the queue, such that

$$x_\Delta \in \boldsymbol{x}_N \quad \& \quad x_\Delta \notin \boldsymbol{x'}_R \quad \& \quad 0 < \Delta \leq N. \tag{4}$$

let $\boldsymbol{a}_{nM}$ be a set of M attributes associated to a song $x_n \in \boldsymbol{x}_N$:

$$\boldsymbol{a}_{nM} = \{a_{n,1}, a_{n,2}, \cdots, a_{n,m} \mid 0 < m \leq M\}; \tag{5}$$

let $\boldsymbol{s}_M$ be a set of M user-controllable settings, such that each setting $s_m \propto P(a_{n,m} \equiv a_{\Delta,m})$:

$$\boldsymbol{s}_M = \{s_1, s_2, \cdots, s_n \mid 0 < m \leq M \quad \& \quad 0 \leq s_n \leq 1\}. \tag{6}$$

each setting is meant to be set to 0 in order to force variation of the corresponding attribute between two consecutive draws, and 1 to force the attribute not to change. A value of 0.5 emulates random variations of the attribute.

Finally, let $0 \leq \beta \leq 1$ be a *memory* coefficient of the system, indicating to what extent previous draws will influence the next ones; $\beta = 0$ will only consider the comparison between x_Δ and the examined song, while $\beta > 0$ will also consider the previous results, with the limit case of $\beta = 1$, where the comparison is made only with the first song extracted.

4.2 Similarity function

Since all attributes have different nature and units, it is not possible to rely on a single similarity function. Songs attributes can be categorical or scalar values: for example, author and album title are categorical, while BPM, year, and audio features are scalar. In both scenarios, similarity is expected to be either 0 or 1. Specifically, in the former case similarity can be expressed as:

$$a \times b := \begin{cases} 0, & \text{if } a \neq b \\ 1, & \text{if } a = b \end{cases}. \tag{7}$$

In the latter case, scalar values are compared in order to obtain a continuous distance measure, then the discrete similarity value is computed by considering a distance threshold (whose value depends on the attribute type) and setting similarity to 0 if the continuous distance is above the threshold, and to 1 otherwise.

4.3 Weighting function

At each draw, each song $x_n \in \boldsymbol{x}_N$ is paired to a probabilistic weight p_n, so to have a set $\boldsymbol{p}_N$ of values computed as:

$$p_n = \begin{cases} 0, & \text{if } x_n \notin \boldsymbol{x'}_R \\ \beta \cdot p_n + (1-\beta) \cdot \tau_n, & \text{otherwise} \end{cases} \tag{8}$$

with

$$\begin{aligned} \tau_n &= \textstyle\prod_{m=1}^{M} (2 \cdot |s_m + \delta_{n,m} - 1| + \epsilon), \\ \delta_{n,m} &= a_{n,m} \times a_{\Delta,m}. \end{aligned} \tag{9}$$

Here $\delta_{n,m}$ represent the similarity between x_n and x_Δ along the m^{th} attribute, while τ_n can be interpreted as follows.

The most important part in the definition of τ_n is the argument of the product, as it dictates how the attribute a_n influences the probabilistic weight; the term $|s_m + \delta_{n,m} - 1|$ takes the same value of the similarity $\delta_{n,m}$ if the corresponding setting value is $s_m = 1$; it becomes $1 - \delta_{n,m}$ when $s_m = 0$; and it converges to 0.5 regardless of the similarity value as s_m approaches 0.5. The additive term $0 < \epsilon \ll 1$ is a very small value needed to avoid all-zeros in $\boldsymbol{p}_N$, while the multiplication factor of 2 has been introduced in order to produce a neutral contribution for $s_m = 0.5$ and an amplification when needed.

Finally, note that the update of p_n in Eq. (8) is based on the previous value of p_n, with the β parameter acting as a weight influencing the *responsiveness* of the change.

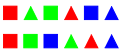
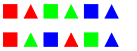
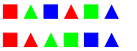
Description	Shape	Color	Output
Random	0.5	0.5	
Alternate	0.0	0.5	
Sort	1.0	0.5	
Interleave	0.0	1.0	
Mix	0.0	0.0	
Group	1.0	1.0	

Table 3: Example of different settings, with 6 colored shapes. β is not considered to simplify the understanding of the base behavior.

4.4 Algorithm and discussion

set $x_\Delta =$ (manual or random choice)
set $\boldsymbol{c}_L = \varnothing$
remove x_Δ from $\boldsymbol{x'}_R$
while $\boldsymbol{x'}_R \neq \varnothing$ && $|\boldsymbol{x'}_R| < L$:
    compute $\boldsymbol{p}_N$
    draw x_α from $\boldsymbol{x}_N$ according to $\boldsymbol{p}_N$;
    add x_α to $\boldsymbol{c}_L$;
    remove x_α from $\boldsymbol{x'}_R$;
    set $x_\Delta = x_\alpha$.

To picture the possible results of this algorithm, consider the case in which *Ruffle* is used to shuffle colored geometric shapes, where the available attributes are color and shape. Starting with a set of 6 elements (3 squares and 3 triangles, forming 3 pairs of colors) the outcomes depicted in Table 3 are possible.

One known drawback of this approach is that all songs that do not manage to fit in the shuffled stream will concentrate at the end of the playlist. This can be resolved by communicating to the user that no more songs are compliant with the settings once the maximum of remaining weights falls under a certain threshold, or can be ignored if the user wants to play the available library completely.

Another possible drawback is that low values of the product terms in τ_n (i.e. close to 0) have a stronger influence when multiplied together than high values (i.e. close to 2). Future implementations will consider a logarithmic type of function for the computation of τ and will assess if it can actually improve the outcomes.

A future evolution could be the generalization of the weight function to non-binary similarity values, in order to take advantage of the continuous nature of scalar attributes and provide more sensitivity to the algorithm.

Finally, a deterministic version of Ruffle is possible, if the draw in the while loop is done by looking for $\text{argmax}_n(\boldsymbol{p}_N)$, nevertheless this case is out of the scope of this paper.

4.5 Possible optimizations

In Eq. (8), and in the pseudo-code above, the computation of $\boldsymbol{p}_N$ is carried out on all $\boldsymbol{x}_N$ songs just for the sake of clarity, of course the computation of $\boldsymbol{p}_N$ can be restricted only to those songs in $\boldsymbol{x'}_R$, without loss in generality.

Beside this, note that since the present work focuses on the validation of the algorithm in its original form, no heuristic optimizations were implemented. Nevertheless some suggestions are provided in the remainder of this section.

Since Eq. (8) is computed ${}^{N^2+N}/_2$ times, the complexity of the algorithm ends up to be $\mathcal{O}(N^2)$. Of course this is not ideal for large music libraries.

In case of $\beta = 1$ (i.e. only the first song is considered), there is no need to compute weights more than once, thus reducing the complexity to $\mathcal{O}(N)$.

Aside from this special case, other improvements may be implemented by storing lookup tables for categorical features such as genre, artist, album etc. even if this does not strictly reduce complexity.

An interesting heuristic may amount to initially shuffle the list with Fisher-Yates, then compute the weights only until a song's weight exceeds a certain threshold. The threshold can be computed as slightly less than the maximum weight possible, which is the product of all $\boldsymbol{s}_M$, in that case the song is picked for next play. If no song reaches the threshold, weights are ready for a regular draw. This heuristic is not ideal for low values of β, since in this case weights are not guaranteed to be up to date.

An alternative to the previous heuristic (assuming that $\beta \ll 1$) is to shuffle the list with Fisher-Yates, then compute the weights only for a fixed number of songs following the one played, and limit the draw in this sliding window.

Finally, consider that real time is not a constraint of the algorithm, since weights can be updated while the user is listening to the song.

5. ASSESSMENT

To assess the validity of *Ruffle* as a usable shuffling algorithm, it has been implemented as a tool to shuffle Spotify playlists, in order to let users evaluate the algorithm before answering a survey.

5.1 Implementation

The tool is based on the Spotify Web API, and is implemented using the vue.js[4] framework. After logging in to their Spotify account, users are able to see the list of their saved tracks, together with a section devoted to load single playlists' content. Using the Web API, for every track the system retrieves 18 properties: 1. track title, 2. artists, 3. album title, 4. genres (associated to artists), 5. release year, 6. duration, 7. key, 8. mode, 9. time signature, 10. acousticness, 11. danceability, 12. energy, 13. instrumentalness, 14. liveness, 15. loudness, 16. speechiness, 17. valence, and 18. tempo. Users can see the values of these properties in the track details. The properties are automatically computed by Spotify, and there is no control or direct knowledge over the algorithms that are implemented for this task.

A sidebar is used to change the settings, i.e. the β and the s_m values, with a set of sliders; a number of presets are available, representing the 5 most selected clusters of Table 2, together with 2 configurations never chosen in previous tests.

[4] https://vuejs.org/

After choosing a preset or manually adjusting single values, users can run the *Ruffle* algorithm, obtaining a reordering in the list of tracks.

The re-ordered tracks can be also saved in a new playlist.

5.2 Evaluation survey

The survey is composed of 4 main parts: a briefing section, containing information about *Ruffle*, and instructions on how to use the implemented Spotify Playlist Shuffler. Please note that it was explicitly asked to focus the attention to the shuffling outcomes rather than the usability of the prototype, since the latter is not the focus of the paper.

The second part repeats some of the questions of the first survey aimed at describing the sampled population, i.e. sex, age, average daily music listening time, usage of shuffle function and satisfaction about the currently used shuffle.

The third part focuses on general impressions on the *Ruffle* algorithm, asking questions about the usefulness of: the available properties, the algorithm itself beside available properties, and the β parameter. Users were also asked to evaluate how much they liked *Ruffle*, and if they would use it if implemented in a music player.

In the last part users were asked to evaluate how frequently they would use each of the presets present in the prototype (note that the "true randomness" preset can be considered as a baseline, since it is equivalent to Fisher-Yates), they were also asked to enter the preferred settings they experienced, and if they would likely change settings frequently, use a finite (small) set of presets, or just use a "set and forget" approach. Finally they were asked to enter free comments if they had any.

The last two parts were aiming at validating what has been observed in Section 3 and evaluating the acceptance of the *Ruffle* algorithm.

5.3 Results

In total 23 users were tested. This survey and the one described in Section 3 have been administered to two different populations, nevertheless, the distributions of answers to the first section were substantially similar.

The algorithm itself (beside available properties) has been marked as useful 17 times, not useful 1 time, and neither useful nor useless 5 times. Similarly the Beta parameter has been marked as useful 17 times, not useful 1 time, and neither useful nor useless 5 times. Furthermore, 19 users liked *Ruffle* (versus 1 that did not liked the algorithm, and 3 which were neutral about it), and 22 over 23 said they would use it in real applications, thus demonstrating that *Ruffle* is indeed a desirable feature. The properties marked as useful are reported in Table 4, together with the number of votes received.

Properties such as *theme* (intended as "Christmas songs" etc.), *instruments*, *mood*, and *language* were manually inserted by subjects.

This not only provides a detailed view of what has been observed in Section 3, but also refines results in a real world scenario, since answers to the preliminary test were given in an hypothetical scenario. In particular it can be seen which audio features are more interesting for users, sometimes even more than metadata.

Attribute	Votes	Attribute	Votes
Artists	18	Loudness	5
Genres	14	Mode	5
Danceability	13	Time Signature	3
Energy	13	Valence	3
Instrumentalness	13	Duration	2
Album Title	9	Language	2
Release Year	9	Liveness	2
Tempo	7	Instruments	1
Speechiness	6	Mood	1
Acousticness	5	Theme	1
Key	5		

Table 4: Properties marked as useful by *Ruffle* users.

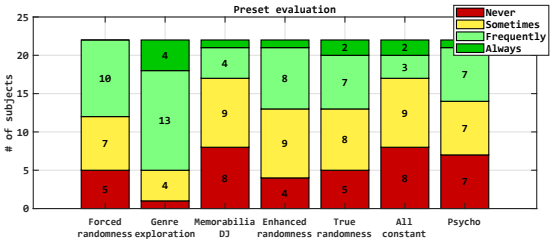


Figure 2: Answers to the question "Would you use this preset in your shuffling sessions?"

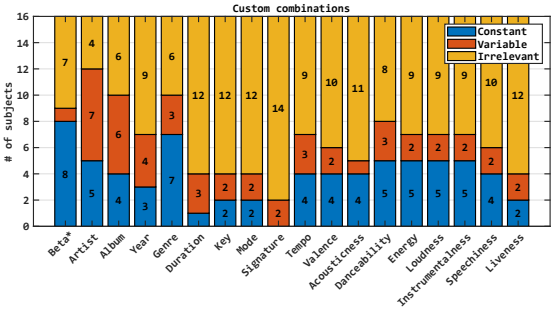


Figure 3: Composition of reported custom presets (for β only, blue ≈ 1, yellow ≈ 0.5, red ≈ 0).

As reported in Figure 2, the most cited configurations of Table 2, when selected by users, revealed to be perceived slightly differently from the expectations. In particular, only "Genre Exploration" clearly outperformed the 2 presets expected to be disliked, which indeed were.

From another perspective, Figure 3 shows how the preferred settings are distributed. This picture highlights how the expressed preferences differed from the "Genre Exploration" preset. This seems to contrast with the outcome of the previous question, but it is worth stressing that the presented presets were working only on 5 principal metadata (Genre, Artist, Album, BPM, and Release Year), while the manual settings were those made available by Spotify and previously listed. This may suggest that, in case of very basic settings, "Genre Exploration" is the most useful approach; nevertheless, when the possibility to finely tune preferences is offered to the user, system customization is a very appreciated feature.

Unfortunately, since the reporting of a preferred personal

setup was optional, only 12 answers were collected, not enough to perform a clustering. Nevertheless, it is interesting to see that all answers were unique, reporting different combinations of preferences.

Concerning the preferred style of settings, the need of only one preset to be "set and forget" has been chosen 5 times, the need for fine tuning each session has been chosen 5 times, and the need for a finite set of custom presets 13 times, thus being the most desirable scenario.

Among free user comments, it is worth mentioning the request to be able to select the first song for the playlist. This aspect, already considered in the discussed version of *Ruffle*, will be soon implemented in the online prototype, too.

Other remarks concerned control values: between 0 and 0.5, they have been perceived as not very useful, and difficult to discern, while they become more significant in the upper part of the scale. In this case, an exponential parameter control may solve the issue: the lower part would be compressed in less space, leaving the upper part more sensitive to changes. In terms of GUI, if the parameter is interpreted as the "amount of homogeneity", a slider in the upper bound position recalls steadiness, while randomness is expected to be closer to the lower bound (with the mid position being considered a midpoint between randomness and steadiness).

6. CONCLUSIONS AND FUTURE WORKS

This paper presented a novel algorithm to shuffle music playlists by giving the user the possibility to configure the dimensions to consider in the calculation of pieces similarity. On each dimension, feature values can be ignored (thus not influencing the process), be distanced as much as possible, or conversely act as piece aggregators.

In order to let the reader test the algorithm, a solution publicly available via web has been released. Such a framework, working on Spotify personal playlists, is available at `http://ruffle.lim.di.unimi.it/`.

Concerning future work, we are planning to perform a fine tuning of the formulas employed for scalar values such as year, tempo, etc. Besides, we aim to implement some features that may improve user's experience, such as the possibility to set the size of the generated playlist (e.g., a shuffled list made of n songs or lasting m minutes).

7. REFERENCES

[1] M. G. Quiñones, "Listening in shuffle mode," *Lied und populäre Kultur/Song and Popular Culture*, pp. 11–22, 2007.

[2] T. W. Leong, F. Vetere, and S. Howard, "The serendipity shuffle," in *Proceedings of the 17th Australia conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future*, 2005, pp. 1–4.

[3] D. Powers, "Lost in the shuffle: Technology, history, and the idea of musical randomness," *Critical studies in media communication*, vol. 31, no. 3, pp. 244–264, 2014.

[4] K. R. M. Sanfilippo, N. Spiro, M. Molina-Solana, and A. Lamont, "Do the shuffle: Exploring reasons for music listening through shuffled play," *PLOS ONE*, vol. 15, no. 2, pp. 1–21, 02 2020. [Online]. Available: https://doi.org/10.1371/journal.pone.0228457

[5] M. Bar-Hillel and W. A. Wagenaar, "The perception of randomness," *Advances in applied mathematics*, vol. 12, no. 4, pp. 428–454, 1991.

[6] R. Croson and J. Sundali, "The gambler's fallacy and the hot hand: Empirical data from casinos," *The Journal of Risk and Uncertainty*, 2005.

[7] G. Smith, *Standard deviations: Flawed assumptions, tortured data, and other ways to lie with statistics*. Abrams, 2014.

[8] O. Celma, *Music recommendation and discovery*. Springer, 2010.

[9] R. A. Fisher and F. Yates, *Statistical tables: For biological, agricultural and medical research*. Oliver and Boyd, 1938.

[10] R. Durstenfeld, "Algorithm 235: random permutation," *Communications of the ACM*, vol. 7, no. 7, p. 420, 1964.

[11] D. Knuth, "Seminumerical algorithms," *The art of computer programming*, vol. 2, 1981.

[12] G. P, G. Gupta, S. H, S. Naseera, A. A. Siddiqui, G. G, G. B. Amali, and S. Gonjari, "Music playlist manager using fisher-yates shuffling algorithm and sorting," *World Wide Journal of Multidisciplinary Research and Development*, 2017.

[13] M. Fiedler, "The art of shuffling music," http://keyj.emphy.de/balanced-shuffle/, 2007, online; accessed 11 March 2021.

[14] P. Singh, A. Batheja, and A. Chowdhury, "Predictive music shuffling algorithm," *International Journal of Computer Science and Information Technologies*, vol. 6, 2015.

[15] R.-H. Liang and Z.-S. Liu, "Towards responsive music shuffling," https://www.researchgate.net/publication/228789429_Towards_Responsive_Music_Shuffling, accessed: 2020-10-14.

[16] L. Poláček, "How to shuffle songs?" https://engineering.atspotify.com/2014/02/28/how-to-shuffle-songs/, 2014, online; accessed 11 March 2021.

[17] M. Castelluccio, "The music genome project," *Strategic Finance*, pp. 57–59, 2006.

SOUND AND MUSIC COMPUTING USING AI: DESIGNING A STANDARD

Marina BOSI[1], **Niccolò PRETTO**[2], **Michelangelo GUARISE**[3], and **Sergio CANAZZA**[2]

[1] *Center for Computer Research in Music and Acoustics (CCRMA)*, **Stanford University**, Stanford, USA
[2] *Centro di Sonologia Computazionale (CSC)*, **University of Padova**, Padua, IT
[3] **Volumio**, Florence, IT

ABSTRACT

While there are currently various approaches that define and adapt the conditions in which the user experiences content or service for several music and audio-related applications including entertainment, communication, audio documents preservation/restoration, we are missing worldwide accepted standards that enable data exchange and interoperability based on common interfaces for such applications. The Moving Picture, Audio and Data Coding by Artificial Intelligence (MPAI) is an international non-profit organization whose mission is to develop such standards. Relying on Artificial Intelligence (AI), MPAI creates a workflow of AI Modules (AIM) that are interchangeable and upgradable without necessarily changing the logic of the application. A specific area of work, MPAI Context-based Audio Enhancement (MPAI-CAE), is showing tremendous possibilities for the Sound and Music Computing (SMC) community. MPAI-CAE applies context information to the input content to deliver the audio output via the most appropriate protocol. Three MPAI-CAE case studies particularly relevant for the SMC community will be presented in this paper: Audio recording preservation (ARP), a use case that covers the whole "philologically informed" archival process of an audio document, from the active sound documents preservation to the access to digitized files; Audio-on-the-go (AOG), which aims to improve safety and listening quality for situations in which the users are in motion in different environments; and Emotion-enhanced speech (EES), a use case that implements a user-friendly system control interface that generates speech with various levels of emotions.

1. INTRODUCTION

Global-scope standardization projects not only offer an indication of the development of an industry but also allow for the partitioning of complex systems into components that can be provided by different sources. This, for example, was the case of media standards in the 1990's. While different companies developed prototypes, the international MPEG standard [1–3] was the catalyst that started a revolution in audio and media consumption. Similarly, MPAI is an international body with the mission to develop standards for data coding that have AI as its core technology. By data coding we mean the transformation of data preserving the semantic aspects that are important to a specific application.

Officially constituted on Wednesday 30 September 2020, MPAI has already produced a considerable body of work. After a Call for Technologies (CfT) for the general AI Framework (MPAI-AIF), MPAI established a Development Committee (MPAI-AIF DC) that is selecting technologies and is currently in the standard development phase (see also Section 2). The goal of the MPAI-AIF standard (see also Section 2 and Figure 1) is to enable the creation and automation of mixed Machine Learning (ML), AI, and legacy data processing modules (collectively AIMs) and to define their use as part of inference workflows. Innovations in AI have led to implementations that can now be found in a wide range of application areas including Speech, Audio, and Image Processing and Recognition. MPAI aims to identify and define interfaces to such implementations to make them usable across as many domains as possible.

MPAI-CAE defines the use of AI to improve the user experience for several audio and music-related applications including entertainment, communication, teleconferencing, gaming, post-production, restoration, etc. The currently available solutions, which adapt to various conditions in order to improve the ultimate user's experience, tend to be vertically integrated. Therefore it is difficult to re-use possibly valuable AI-based components for different applications and different platforms. MPAI-CAE intends to define interfaces between distinct stages (AIMs) to promote the development of horizontal markets of competing solutions tapping into and further promoting AI innovation. Adopting AIMs that are reusable, updatable and extensible, MPAI-CAE intends to define standards for AIM interfaces (i.e., input and output format) but is silent as to the AIM internals. Therefore, the performance of AIMs can continuously improve by incorporating new technologies. The performance evaluation process is under development in a separate standardization thread involving all the MPAI standards and it will be discussed in further work.

The potential impact on the SMC community is huge. MPAI-CAE will allow researchers to carry out sophisticated optimizations that enable a superior user experience. Such optimizations can then be implemented in AIMs by third party providers. Manufacturers and service providers

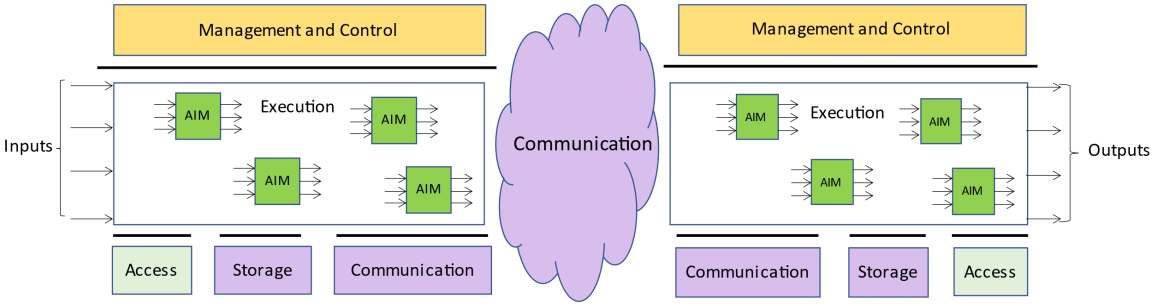


Figure 1. AI Framework schema.

can subsequently adopt these optimized AIMs in their products and services. In general, MPAI operates based on an open international collaboration of interested parties who support the MPAI mission and the means to accomplish it. The use cases described in this paper will help illustrate the core of the MPAI-CAE effort.

The remainder of the manuscript is organized as follows. Section 2 reviews the basic structure of the MPAI process, whereas Section 3 offers a detailed description of three MPAI-CAE use cases particularly relevant for the SMC community (ARP, Section 3.1; AOG, Section 3.2; and EES Section 3.3). Section 4 concludes the paper.

2. STRUCTURE

The overall structure of MPAI relies on MPAI-AIF. The smallest units are the AIMs which are computational modules trained for specific tasks by exploiting AI, ML, and legacy data processing and that can be implemented in hardware, software and mixed hardware/software. MPAI does not define the internal behavior of the AIMs, nevertheless clearly specifies the syntax and semantics of the interfaces. AIMs operate in the standard AI framework and exchange data in specified formats. For this reason, AIMs are replaceable, re-usable and upgradable without changing the logic of the application, fostering the continuous improvement of the AI technology. Different AIMs can be seamlessly interconnected as in the examples provided in the next Sections.

The framework can create, compose, execute, and update multi-vendor AIMs. As can be seen in Figure 1, the Framework is composed by six main components: (a) *Management and Control* manages and controls the AIMs; (b) *Execution* is the environment in which combinations of AIMs operate; (c) *AIMs*, already described; (d) *Communication* is the basic infrastructure used to connect possibly remote Components and AIMs; (e) *Storage* encompasses traditional storage; (f) *Access* represents the access to static or slowly changing data that are required by the application.

The standardization process of MPAI follows an approach based on seven stages. The initial stage (stage 0) concerns the gathering of interest in developing use cases related to a specific topic. As the use cases are not part of the normative standard, they can be augmented later in the process. The information collected in the stage 0 is formalized in the 1^{st} stage. In this stage, the use cases are characterized and a detailed work plan is delineated. The 2^{nd} stage consists in the definition of the functional requirements for a specific work area. The 3^{rd} stage finalizes the commercial requirements, specifically the development of the framework license. The 4^{th} stage formalizes the CfT while the 5^{th} promotes the development of the standard. During the 6^{th} and last stage the standard is approved and published. As of today (March 14, 2021), MPAI-AIF is in the 5^{th} stage, while the MPAI-CAE project is in the 4^{th} stage.

From the bottom-up approach described above, a wide variety of high-tech schemes related to current hot topics in AI are developed. In this context, MPAI-CAE is one of the most promising areas of work. Three of the main MPAI-CAE use cases will be described in the next section.

3. USE CASES

3.1 Audio Recording Preservation (ARP)

MPAI-CAE covers several different SMC areas. The ARP use case represents an important example in the preservation of open-reel analog audio tapes. As shown in Figure 2, the input of this use case is the audio of a digitized tape and the video of that tape flowing on the magnetic head of the tape recording as described in [4].

The first module is the *Audio Enhancer*. This is an optional module consisting of a "denoiser" (in a broad sense). This stage compensates for eventual errors caused by misaligned recording equipment and/or for tape hiss caused by the imperfections introduced by aging (see Storm's Type B, that defines a historically faithful level of reproduc-

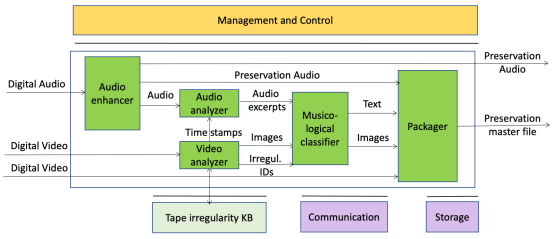


Figure 2. Audio Recording Preservation workflow.

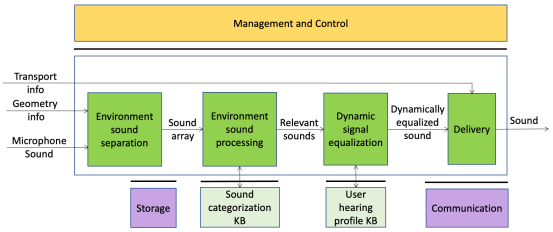


Figure 3. Audio-on-the-go workflow.

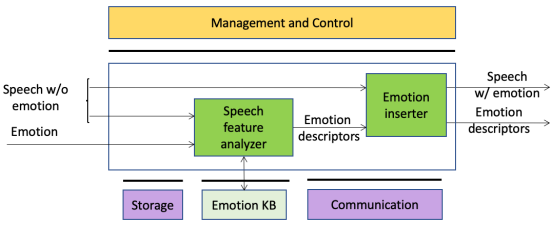


Figure 4. Emotion enhanced speech (EES) workflow.

tion [5]).

The video input is analyzed by an AI algorithm for detecting irregularities (splices, damages, marking, etc.) on the tape such as [4]. This module could interact with the external *Tape Irregularity Knowledge Base (KB)*. Its output consists of frames of the irregularities extracted from the video, their related ID, and the timestamp related to the irregularity. The video of the tape also includes a low quality audio that can be used to synchronize the high quality audio stream with the video itself. Relevant audio excerpts corresponding to the detected irregularities can be extracted and analyzed by the *Audio Analyzer* module.

Single frames concerning irregularities and the corresponding audio excerpts are then analyzed by the *Musicological classifier* [1] that aims to select and describe relevant irregularities. The resulting description and images will be part of the preservation master file created by the *Packager* module. Therefore the preservation master file, composed by audio, video, metadata as indicated in [6], will be provided as output.

In addition, an alternative output is provided. It consists of the digitized tape audio that could be used for accessing the audio content without using the preservation master.

3.2 Audio-on-the-go (AOG)

MPAI-CAE Audio-On-The-Go is a use case that aims to improve safety and listening quality in various situations in which users are on the move, like in a car, with a bike, running and so on. For example while biking in the middle of city traffic, the user should enjoy a satisfactory listening experience without losing contact with the acoustic surroundings. There will be sounds which are not relevant for safety (like wind noise) and sounds which are (like the horn of a car or incoming traffic), and therefore such sounds shall be selected and presented to the user only if relevant for safety [7]. This is achieved thanks to the microphones available in earphones and earbuds capturing the signals from the environment, the relevant environment sounds (i.e., the horn of a car) are then selectively recognized. In addition, the sound rendition is adapted to the acoustic environment, providing an enhanced audio experience (e.g., performing dynamic signal equalization) and allowing a more energy efficient operation resulting in an improved battery life. In this use case, the goal is achieved by using a series of AIMs. The first AIM (*Environmental Sound Separation*) is fed with Microphone sound which captures the surrounding environment noise, together with according geometry information (which describes number, positioning and configuration of the microphone or the array of microphones). The sounds are then categorized following prescriptions of a *Sound Categorization Knowledge Base* (queried by the corresponding AIM), resulting in a sounds array and their categorization. Sound samples might eventually be compressed to allow a cloud-processing procedure. The *Environmental Sound Processing* AIM, after fetching a list of relevant sounds from a KB, will trim sounds not relevant for the user in the specific moment and feed them to the next AIM, *Dynamic Signal Equalization*. This AIM fetches the *User Hearing Profile from a Knowledge Base* and equalizes dynamically the sound taking into account the user's specific hearing deviations. Finally, the resulting sound is delivered to the output via the most appropriate the *Delivery* method, such as Bluetooth 5.0 or any compatible protocol.

[1] The use of the term "musicological classifier" was selected because it specifically identifies a classification of interest for musicologists.

3.3 Emotion enhanced speech (EES)

Speech carries information not only about the lexical content, but also about a variety of other aspects such as age, gender, signature, and emotional state of the speaker, and this is an acknowledged issue. Speech synthesis is evolving towards supporting these aspects. There are many cases where a speech without emotion needs to be converted to a speech carrying an emotion, possibly with grades of a particular emotion. This is the case, for instance, of a human-machine dialogue where the message conveyed by the machine is more effective if it carries an emotion properly related to the emotion detected in the human speaker. MPAI-CAE EES use case aims to standardize a *natural* communication by virtual agents, and thus improve the quality of human-machine interaction, by making it closer to a human-human interaction (e.g., [8,9]). By means of EES anyone can realize a user-friendly system control interface that lets users generate speech with various — continuous and real-time — expressiveness control levels.

The MPAI-CAE EES can be implemented as in figure 4, using data processing technology or artificial intelligent technology, where a neural network incorporates the *Emotion Knowledge Base* information.

The inputs are: a *neutral* (without specific emotion) speech, synthesized or recorded; a text file with the annotation of which basic emotion [10,11] to insert (and where) into the speech signal.

The *Speech feature analyzer* extracts the speech features, queries the *Emotion KB* and obtains *Emotion descriptors* (a subset of speech features modified accordingly to the particular emotion). Alternatively, *Emotion descriptors* are produced by an embedded neural network.

Emotion Knowledge Base exposes an interface that allows *Speech feature analyzer* to query a KB of speech features extracted from recordings of different speakers reading/reciting the same corpus of texts, with the standard set of basic emotions and without emotion, for different languages and genders. A set of acoustic cues are used to compare the voice quality characteristics of the speech signals on a voice corpus in which different emotions are reproduced. The psychoacoustic parameters of emotions in speech can be separated into two groups [12]: prosodic (rhythm, speed of speech, intonation and intensity) and vocal frequency-related parameters (timbre, fundamental tracking, position of the formants and distribution of the spectral energy).

Emotion inserter inserts a particular emotional vocal timbre, e.g., anger, disgust, fear, happiness, sadness, and surprise into a neutral (emotion-less) synthesized voice. It also changes the strength of an emotion (from neutral speech) in a gradual fashion.

4. CONCLUSIONS

A group of highly motivated experts in different fields has gathered within the MPAI community (`https://mpai.community/organisation/`) to develop use cases aggregated in areas where the MPAI standards can have a big impact. Thanks to the efforts of many, MPAI has reached several important milestones. For example, MPAI-AIF has already reached the standard development stage and multiple areas, including MPAI-CAE, have open call for technologies.

This paper presented three use cases of MPAI-CAE that are of particular interest to the SMC community. Other MPAI areas of work include Multi-modal conversation, AI-Enhanced traditional video coding, integrative AI-based analysis of multi-source genomic/sensor experiments, Compression and understanding of financial data, and Server-based predictive distributed multiplayer online gaming.

MPAI has introduced a number of innovative approaches in both the technologies that address specific industries and also in the development of licensing guidelines. MPAI is planning to develop for each standard a "framework licence" to overcome the ambiguities of the Fair, Reasonable and Non-Discriminatory (FRAND) model. Such framework licenses are already available for MPAI-AIF, MPAI-MMC and MPAI-CAE.

Moreover, MPAI pledges to address ethical questions raised by its technical work and is in the process of defining different procedures in this area.

Acknowledgments

The authors would like to thank the MPAI community for the invaluable discussions and, in particular, Leonardo Chiariglione, MPAI President, for his vision and his continued guidance.

5. REFERENCES

[1] ISO/IEC, "Coding of moving pictures and associated audio for digital storage media at up to about 1.5 mbit/s," *ISO/IEC 11172 - International Organization for Standardization/International Electrotechnical Commission and others*, 1993.

[2] ——, "Information technology — generic coding of moving pictures and associated audio information," *ISO/IEC 13818:1995 - International Organization for Standardization/International Electrotechnical Commission and others*, 1995.

[3] K. Brandenburg and M. Bosi, "Overview of MPEG audio: current and future standards for low bit-rate audio coding," *Journal of the Audio Engineering Society*, vol. 45, no. 1/2, pp. 4–21, 1997.

[4] N. Pretto, C. Fantozzi, E. Micheloni, V. Burini, and S. Canazza, "Computing methodologies supporting the preservation of electroacoustic music from analog magnetic tape," *Computer Music Journal*, vol. 42, no. 4, pp. 59–74, 2019, doi: 10.1162/comj_a_00487.

[5] F. Bressan and S. Canazza, "A systemic approach to the preservation of audio documents: Methodology and software tools," *Journal of Electrical and Computer Engineering*, 2013, doi: 10.1155/2013/489515.

[6] C. Fantozzi, F. Bressan, N. Pretto, and S. Canazza, "Tape music archives: from preservation to access," *International Journal on Digital Libraries*, vol. 18, no. 3, pp. 233–249, 2017, doi: 10.1007/s00799-017-0208-8.

[7] B. Schulte-Fortkamp, "Soundscape, standardization, and application," in *Proc. Int. Conf. Euronoise 2018*, Crete, 2018, pp. 2445–2450.

[8] J. E. Cahn, "The generation of affect in synthesized speech," *Journal of the American Voice I/O Society*, vol. 8, pp. 1–19, 1990.

[9] M. A. M. Shaikh, A. R. F. Rebordao, K. Hirose, and M. Ishizuka, "Emotional speech synthesis by sensing affective information from text," in *2009 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops*, 2009, pp. 1–6.

[10] R. Plutchik, "A psychoevolutionary theory of emotions," *Social Science Information*, vol. 21, no. 4-5, pp. 529–553, 1982. [Online]. Available: https://doi.org/10.1177/053901882021004003

[11] T. Dalgleish and M. J. Powers, *Handbook of Cognition and Emotion*. Wiley, 1999.

[12] N. Tits, "A methodology for controlling the emotional expressiveness in synthetic speech-a deep learning approach," in *2019 8th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW)*. IEEE, 2019, pp. 1–5.

IS AN AUDITORY EVENT MORE TAKETE?

Federico FONTANA (federico.fontana@uniud.it) (0000-0002-1692-2603)[1],
Hanna JÄRVELÄINEN (hanna.jarvelainen@zhdk.ch) (0000-0001-6255-8657)[2], and **Maurizio FAVARO**[1]

[1]*Department of Mathematics, Computer Science and Physics (DMIF)*, **University of Udine**, Udine, Italy
[2]*Institute for Computer Music and Sound Technology (ICST)*, **Zurich University of the Arts**, Zürich, Switzerland

ABSTRACT

Recent experiments have demonstrated that the words Takete and Maluma, as well as Kiki and Bouba, once heard stimulate a cross-modal response in humans that goes beyond visual associations, and in particular affects the trajectory of human motion patterns. Inspired by such experiments, in a binary (Takete/Maluma) response test we presented to sixteen individuals a random sequence of either sonic or silent videos reproducing a smooth and a notched ball rolling down along a rounded or zig-zagged path. Bayesian estimation revealed a credible effect of the zig-zagged path in participants choosing Takete, and an equally strong effect of the notched ball. On the other hand, the silent videos had a negative effect on subjects' probability of choosing Takete. This means that in absence of auditory feedback, subjects tend to choose Maluma compared to similar situations with sound. Though exploratory, such a result suggests that the auditory modality may have significantly biased the decision toward Takete when our participants were exposed to the audio-visual event. If supported by more extensive tests, this experiment would emphasize the importance of sound in the cognition of audio-visual events eliciting sense of sharpness in humans.

1. INTRODUCTION

In 1929, Wolfgang Köhler asked a group of Spanish speakers to make an association between the words Takete or Maluma and the images of two shapes, one jagged and the other rounded, like those in Figure 1.

His results showed a significant preference of the speakers for associating Takete with the jagged, and Maluma with the rounded shape. The experiment has been repeated by several psychologists using different pairs of words, in particular Kiki and Bouba, as well as involving speakers from different languages and levels of literacy. Apart from some specific exceptions reported for a population of Papua New Guinea, these experiments have all shown a general tendency of speakers, including young children aged 2.5 years old [1], to map rounded shapes on words containing the vowels "o" and "u", and, conversely, jagged shapes on words containing "e" and "i".

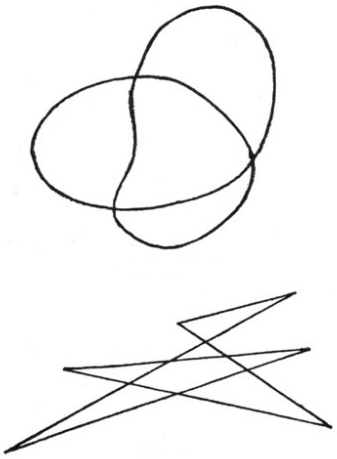


Figure 1. Images similar to those used by Köhler in his experiment.

Taken together, these results provide evidence of a powerful cross-modal effect, linking visual shapes to sounds of words. The presence of this effect in pre-literate children suggests the existence of active connections among contiguous cortical areas, making possible for humans to link characteristic geometrical shape contours to similar geometries assumed by the speaker's lips. According to Ramachandran and Hubbard [2] such connections exist before language, hence they represent a general invariant speeding up and constraining its development.

This research embraced other sensory modalities in more recent decades, investigating associations that are not dominated by vision. Spence and colleagues investigated effects of taste [3, 4]: by asking subjects to associate food and liquids to Kiki/Bouba, they concluded that counter-intuitive branding and packaging may be detrimental to the success of a food product. Similar effects were found for odors [5]. A stronger input to our work, however, comes from experiments that involved motion patterns. Independently of each other, in 2016 Shinohara *et al.* and Koppensteiner *et al.* presented an experiment linking gestures to Takete/Maluma by using animated human figures [6, 7]. Earlier in 2013, Fontana had experimented on the same link by guiding the dominant hand of blindfolded partici-

pants along rounded or jagged trajectories by means of a robotic arm, hence excluding vision completely from the tests [8]. Together, these experiments shed further light on the human ability to associate words to sensations involving motion. Further considerations about this ability and its relationships with previously memorized mental imageries were pointed out by Fryer *et al.* while testing haptic-word associations made by blind individuals [9], and then rediscussed by Graven & Desebrock [10].

Sensation of motion becomes unavoidable if an experiment is designed involving auditory stimuli. Familiar sounds in fact are almost inevitably linked to dynamic events, in which motion is inherently implied. However, the association between words and auditory stimuli is a fragile concept by definition. As words encode sounds, an experiment of this kind should first provide evidence that the association between an auditory stimulus and the sound of a word is not merely onomatopoeic. Probably due to this issue, that puts the own concept of *association* under discussion, experiments linking auditory feedback to words are apparently absent in the literature. However, two ideas convinced us to proceed along this uneven path:

- if the sound of our words of interest is a consequence of onomatopoeia [2], then auditory stimuli should be chosen among familiar sounds that do not imply the words Takete/Maluma or Kiki/Bouba via an evident onomatopoeic link, as e.g. a tik-tok or mumbling sound would suggest;

- if auditory feedback is able to define a genuine, that is, not onomatopoeic association with such words, then the effect can be controlled by removing sound from a multi-sensory stimulus in which this feedback is superimposed as part of a multi-modal event presentation.

Moved by such ideas, we designed an experiment in which participants had to classify a ball rolling down as Takete or Maluma. Two audio-visual components were present in each stimulus: the ball surface and the path trajectory. The surface was either smooth or notched; the trajectory was either rounded or zig-zagged. In what follows, the smooth/round conditions are marked with M (Maluma), and the notched/zig-zagged conditions are marked with T (Takete) according to their respective hypothesized association. Holding such two visible differences, the corresponding rolling sounds of the two balls and the collision sounds they did against the side walls while traversing the respective paths were different as well.

2. METHOD

With all laboratories at the university being inaccessible to students and guests due to the covid pandemic, the experiment took place in a quiet room at one of the Authors' home.

Figure 2. Balls (above) and paths (below) used in the experiment.

2.1 Participants

Sixteen participants (8 female and 8 male, ages M=40.6, sd=17.6 years), all reporting normal sight and hearing volunteered for the experiment. Two of them reported previous knowlwedge of the Köhler experiment.

2.2 Setup and stimuli

Two balls were made of white play dough covered with vinyl glue (Figure 2, above), both having an external diameter of about 6 cm and a weight of about 250 g. In parallel, two paths were prepared on a plywood base sized $1\times0.5\times0.15$ m (Figure 2, below), again using play dough covered with vinyl glue for the side walls delimiting the paths. Once such paths were refined so as to provide an approximately identical time to reach the bottom, the side walls were secured to the base with permanent glue and the setup was painted. A contrast between dark still and bright moving objects was created, similar to the scenario that Shinohara *et al.* had presented to their participants [6].

The two balls were video- and audio-recorded while they rolled down along both paths, once being left free to roll by one Author who wore a dark cloth. Four short sonic videos hence were recorded, three times each. Each video, then, was duplicated by removing the soundtrack. Twenty-four stimuli, twelve sonic and twelve silent videos, were finally

made available for the tests. [1]

2.3 Procedure

While sitting in front of a PC equipped also with speakers, each participant was asked to attend some videos of what was told to be a simple passtime game popular in Polinesia. Two different versions were told to exist about this game, Takete or Maluma as they were called by locals, and participants had to label each video accordingly when it was finished, by verbally reporting their choice to the experimenter; alternatively they could see it again, by pressing the space bar of the PC keyboard. Once a decision was made, they attended the next video.

The videos were included in a randomly balanced sequence of trials, for a total of 2 balls {T, M} × 2 paths {T, M} × 2 modality {V, AV} × 3 repetitions = 24 trials. Each session lasted about 10 minutes. At the end of it, each participant left comments to the experimenter in particular including information about his or her previous knowledge of this experiment.

3. RESULTS

Results are presented in Figure 3. Inspecting the raw data, it seems that incongruent combinations of ball and path (M-T or T-M) lead to relatively even distributions of Takete and Maluma responses – however, in favor of Takete in audio-video conditions (AV) and in favor of Maluma in the video only conditions (V). In congruent ball-path combinations M-M and T-T, responses are biased toward Maluma and Takete respectively, as expected. Yet again, Takete responses are generally favored in the AV conditions and Maluma in the V conditions, so much so that with T ball and T path, responses based on video only (V) approach random.

Statistical analysis was carried out by logistic regression as explained below. The model coefficients were estimated by Bayesian methods using the R program and the *brms* package [11–13].

The Takete/Maluma response, a binary-outcome dependent variable, was mapped to values $k = 0$ (Maluma response) and $k = 1$ (Takete response). Such an outcome follows the Bernoulli distribution, taking value 1 with unknown probability p and value 0 with probability $1 - p$:

$$f(k;p) = \begin{cases} p & \text{if } k = 1 \\ 1-p & \text{if } k = 0 \end{cases} \quad (1)$$

The unknown probability p of a Takete outcome was predicted by a logistic regression model given by

$$\log \frac{p}{1-p} = \beta_0 + \sum_{i=1}^{m} \beta_i \cdot x_i, \quad (2)$$

where $m = 3$ is the number of predictors, x_i are the predictors (path, ball, and modality; the effect of repetition was not modeled), β_0 is the intercept, and β_i are the regression coefficients estimated by the model.

[1] The videos are available at `https://doi.org/10.5281/zenodo.4770168`.

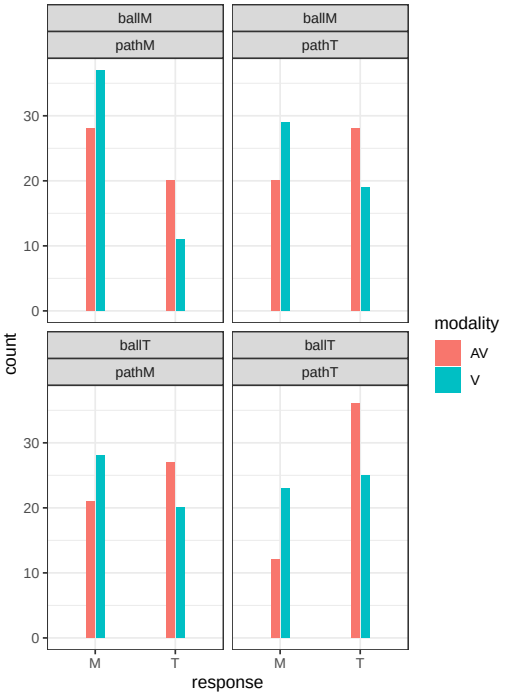


Figure 3. Results. y-axis = response counts (M: Maluma; T: Takete) for both modalities (AV: audio-video; V: video) in each factor combination (ballM: smooth ball; ballT: notched ball; pathM: rounded path; pathT: zig-zagged path).

Figure 4 presents the parameter estimates and their 95% Credible Intervals from the posterior distribution, produced by Markov chain Monte Carlo (MCMC) draws. These logit-transformed values [2] cannot be interpreted in terms of probabilities; however, a 95% CI either entirely above or below zero indicates a credible non-zero positive or negative effect on p, respectively. Hence, Takete path and Takete ball both have an equally strong positive effect. In contrast, video without audio produces credibly more often a Maluma response than video and audio combined. The conditional effects, transformed back to probabilities, are presented in Figure 5.

4. DISCUSSION

This explorative experiment demonstrated that the event of a ball rolling on a surface can be perceived as Takete or Maluma depending on both smoothness of the ball and shape of the trajectory. Our statistical model was additive; modeling the ball-path interaction would require a larger dataset. In a larger experiment, measurement of decision times should also be informative, given that decisions tend to take longer under increasing uncertainty [14, 15]. This could help in investigating, whether either the ball or the

[2] The logit function maps values from $p \in [0, 1]$ to $x \in [-\infty, \infty]$ according to $x = \log(\frac{p}{1-p})$; the inverse mapping is given by the logistic function $p = \frac{1}{1+e^{-x}}$

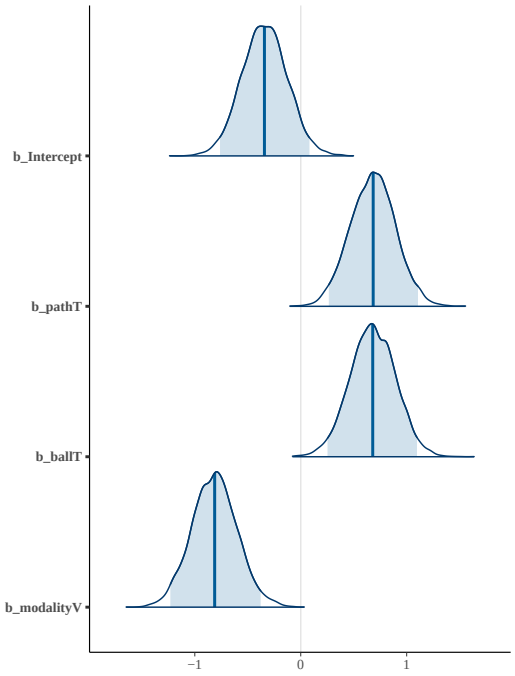


Figure 4. Parameter estimates from the posterior distribution of the Bayesian model.

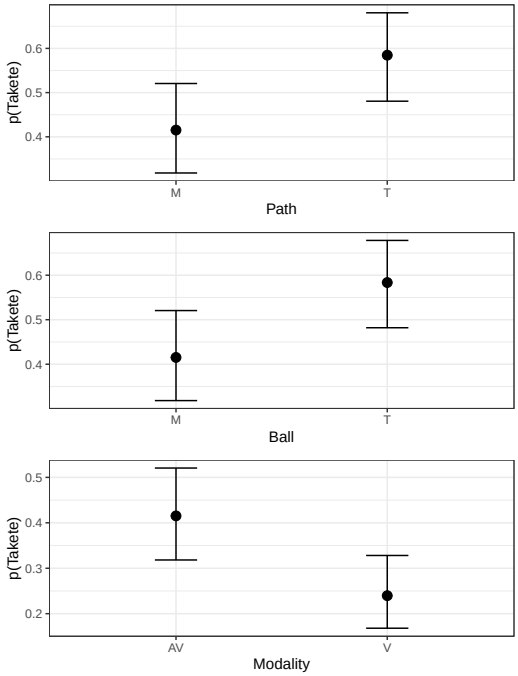


Figure 5. Conditional effects of path, ball, and modality. y-axis = estimated probability of a Takete response; errorbars = 95% Credible Intervals.

trajectory is a dominant feature in the rolling event. In the present model, their effects were approximately equal.

Interestingly, a Takete response was more probable in presence of sound. As the audio and video signals were always congruent, we should expect responses at least in the congruent ball-path conditions (T-T and M-M) to be overwhelmingly in favor of Takete and Maluma, respectively. In both cases however, we see the bias towards Takete when sound is present and towards Maluma in the silent videos. Although our statistical model does not allow very refined conclusions, the raw data suggests that sound is crucial for making non-random decisions in the T-T condition (bottom-right panel in Figure 3). It is of course possible that these specific trajectories or balls happened to produce acoustic cues that were perceived as Takete and visual cues that were perceived as Maluma; using a pseudo-random variety of both might reduce the bias.

Humans (as well as great apes, to some extent) show visual preference for curved objects [16, 17]. In this experiment, most trials contained a curved path or a smooth ball, or both. This might explain part of the Maluma bias in the silent videos, if participants' decisions were guided by higher attention to the pleasant curved components. Auditory information, in contrast, has shown potentially higher alerting power than visual information [18]. Some studies have also reported higher attention to the auditory over the visual channel in high-arousal conditions, although contradictory evidence also exists [19, 20]. Altogether, audiovisual associations to Takete and Maluma are not yet explored in detail.

Regarding the auditory channel, associations of musical excerpts to Takete or Maluma were experimentally found [21], although the related analysis did not explain which factors may have determined the associations.We plan a further experiment, adding an auditory only condition and related signal analysis, to identify cues underlying Maluma and Takete responses. We hope to recruit more participants such that the A/V/AV modalities could be split between subjects. Literature into acoustic cues driving sound-shape symbolism mentions links between angularity, pitch, and other spectral aspects [22]. Material characteristics, such as hardness, might also drive the responses; high importance of auditory cues in identification of materials from bouncing events has been demonstrated [23].

As our data do not yet include the auditory only condition, it is possible that the Takete bias in the AV condition was caused by cross-modal enhancement, similar to the effect observed by Stein et al. [24]. They reported increased visual brightness in presence of an auditory noise burst, although later research has offered other than perceptual explanations for the effect [25].

Assuming that the audiovisual Takete effect was indeed caused by auditory influence, we turn to the question of how the auditory channel may have achieved such dominance; there is ample evidence of general visual dominance, for example in terms of the Colavita effect [26, 27]. In our experiment, the strong auditory influence could be explained, firstly, by higher attention to the auditory channel when sound appears. Attention is known to modulate the visual dominance effect [28]. Secondly, our congruent

stimuli likely increased the importance of auditory cues in feature integration based on the whole-object bias – the spreading of attention to other modalities containing coherent information (see [29]). Auditory dominance has also been demonstrated in situations involving temporal processing [30], or when the auditory channel is more reliable or contains more information, such as music.

5. CONCLUSIONS

Our results showed a credible Takete bias in the audio-visual versus visual condition. We cannot, however, provide a general conclusive answer about the potential of sound to bias a sensation. As we have discussed, the auditory feedback coming from the rolling balls may have biased our participants toward Takete due to specific "associative cues" in those sounds operating above their obvious interpretation, in terms of the physical events they reported about. If existing, such cues are yet to be understood. On the other hand, the suggestions posed by these results motivate the design of further experiments that could contribute to clarifying the role of sound in multi-sensory associations.

6. REFERENCES

[1] D. Maurer, T. Pathman, and C. Mondloch, "The shape of boubas: sound-shape correspondences in toddlers and adults," *Developmental Science*, vol. 9, no. 3, pp. 316–322, 2006.

[2] V. Ramachandran and E. Hubbard, "Synaesthesia: A window into perception, thought and language," *J. of Counsciousness Studies*, vol. 8, no. 12, pp. 3–34, 2001.

[3] A.-S. Crisinel, S. Jones, and C. Spence, "'The sweet taste of Maluma': Crossmodal associations between tastes and words," *Chemical Perception*, vol. 5, pp. 266–273, Aug. 2012.

[4] A. Gallace, E. Boschin, and C. Spence, "On the taste of "Bouba" and "Kiki": An exploration of word-food associations in neurologically normal participants," *Cognitive Neuroscience*, vol. 2, no. 1, pp. 34–46, 2011.

[5] H.-S. Seo, A. Arshamian, K. Schemmer, I. Scheer, T. Sander, G. Ritter, and T. Hummel, "Cross-modal integration between odors and abstract symbols," *Neuroscience Letters*, vol. 478, no. 3, pp. 175–178, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304394010005744

[6] K. Shinohara, N. Yamauchi, S. Kawahara, and H. Tanaka, "Takete and maluma in action: A cross-modal relationship between gestures and sounds," *PLOS ONE*, vol. 11, no. 9, pp. 1–17, 09 2016. [Online]. Available: https://doi.org/10.1371/journal.pone.0163525

[7] M. Koppensteiner, P. Stephan, and J. P. M. Jäschke, "Shaking takete and flowing maluma. non-sense words are associated with motion patterns," *PLOS ONE*, vol. 11, no. 3, pp. 1–13, 03 2016. [Online]. Available: https://doi.org/10.1371/journal.pone.0150610

[8] F. Fontana, "Association of haptic trajectories to takete and maluma," in *Haptic and Audio Interaction Design*, I. Oakley and S. Brewster, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 60–68.

[9] L. Fryer, J. Freeman, and L. Pring, "Touching words is not enough: How visual experience influences haptic–auditory associations in the "Bouba–Kiki" effect," *Cognition*, vol. 132, no. 2, pp. 164–173, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010027714000559

[10] T. Graven and C. Desebrock, "Bouba or kiki with and without vision: Shape-audio regularities and mental images," *Acta Psychologica*, vol. 188, pp. 200–212, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0001691817305176

[11] J. K. Kruschke, *Doing Bayesian data analysis - A tutorial with R, JAGS, and Stan*, 2nd ed. Academic Press, 2014.

[12] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2020. [Online]. Available: https://www.R-project.org/

[13] P.-C. Bürkner, "brms: An R package for Bayesian multilevel models using Stan," *J. Stat. Softw.*, vol. 80, no. 1, 2017. [Online]. Available: http://www.jstatsoft.org/v80/i01/

[14] H. Piéron, "Recherches sur les lois de variation des temps de laten- ce sensorielle en fonction des intensités excitatrices," *L'Année Psychol.*, vol. 20, pp. 17–96, 1914.

[15] ——, *The Sensations*. New Haven, CT: Yale University Press, 1950.

[16] E. Munar, G. Gómez-Puerto, J. Call, and M. Nadal, "Common Visual Preference for Curved Contours in Humans and Great Apes," *PLoS One*, vol. 10, no. 11, p. e0141106, nov 2015. [Online]. Available: https://dx.plos.org/10.1371/journal.pone.0141106

[17] M. Bertamini, L. Palumbo, T. N. Gheorghes, and M. Galatsidas, "Do observers like curvature or do they dislike angularity?" *Br. J. Psychol.*, vol. 107, no. 1, pp. 154–178, 2016.

[18] G. A. D. Souza, L. A. Torres, V. S. Dani, D. S. Villa, A. T. Larico, A. MacIel, and L. Nedel, "Evaluation of visual, auditory and vibro-tactile alerts in supervised interfaces," *Proc. - 2018 20th Symp. Virtual Augment. Reality, SVR 2018*, vol. 2, no. April 2020, pp. 163–169, 2018.

[19] K. L. Shapiro, B. Egerman, and R. M. Klein, "Effects of arousal on human visual dominance," *Percept. Psychophys.*, vol. 35, no. 6, pp. 547–552, nov 1984.

[Online]. Available: http://link.springer.com/10.3758/BF03205951

[20] S. Van Damme, G. Crombez, and C. Spence, "Is visual dominance modulated by the threat value of visual and auditory stimuli?" *Exp. Brain Res.*, vol. 193, no. 2, pp. 197–204, feb 2009. [Online]. Available: http://link.springer.com/10.1007/s00221-008-1608-1

[21] M. Murari, A. Rodà, S. Canazza, G. D. Poli, and O. D. Pos, "Is Vivaldi smooth and takete? Non-verbal sensory scales for describing music qualities," *J. of New Music Research*, vol. 44, no. 4, pp. 359–372, 2015. [Online]. Available: https://doi.org/10.1080/09298215.2015.1101475

[22] K. Knoeferle, J. Li, E. Maggioni, and C. Spence, "What drives sound symbolism ? Different acoustic cues underlie sound-size and sound-shape mappings," *Sci. Rep.*, no. December 2016, pp. 1–11, 2017. [Online]. Available: http://dx.doi.org/10.1038/s41598-017-05965-y

[23] Y. De Pra, F. Fontana, H. Järveläinen, S. Papetti, and M. Simonato, "Does it ping or pong? Auditory and tactile classification of materials by bouncing events," *ACM Transactions on Applied Perception (TAP)*, vol. 17, no. 2, pp. 1–17, 2020.

[24] B. E. Stein, N. London, L. K. Wilkinson, and D. D. Price, "Enhancement of perceived visual intensity by auditory stimuli: a psychophysical analysis." *J. Cogn. Neurosci.*, vol. 8, no. 6, pp. 497–506, nov 1996. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/23961981

[25] C. Spence and M. K. Ngo, "Does attention or multisensory integration explain the cross-modal facilitation of masked visual target identification?" in *The New Handbook of Multisensory Processing*, B. E. Stein, Ed. MIT Press, 2012, ch. 18.

[26] F. B. Colavita, "Human sensory dominance," *Percept. Psychophys.*, vol. 16, no. 2, pp. 409–412, mar 1974. [Online]. Available: http://link.springer.com/10.3758/BF03203962

[27] C. Spence, C. Parise, and Y.-C. Chen, *The Colavita Visual Dominance Effect*. CRC Press/Taylor & Francis, 2012. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/22593876

[28] S. Sinnett, C. Spence, and S. Soto-Faraco, "Visual dominance and attention: The Colavita effect revisited," *Percept. Psychophys.*, vol. 69, no. 5, pp. 673–686, 2007.

[29] I. Fiebelkorn, J. Foxe, and S. Molholm, "Attention and multisensory feature integration," in *The New Handbook of Multisensory Processing*, B. E. Stein, Ed. MIT Press, 2012, ch. 21.

[30] B. H. Repp and A. Penel, "Auditory dominance in temporal processing: New evidence from synchronization with simultaneous visual and auditory sequences." *J. Exp. Psychol. Hum. Percept. Perform.*, vol. 28, no. 5, pp. 1085–1099, 2002. [Online]. Available: http://doi.apa.org/getdoi.cfm?doi=10.1037/0096-1523.28.5.1085

MUSICAL PROSODY-DRIVEN EMOTION CLASSIFICATION: INTERPRETING VOCALISTS PORTRAYAL OF EMOTIONS THROUGH MACHINE LEARNING

Nicholas FARRIS (nicholas.farris@gatech.edu) [1], **Brian MODEL** (bmodel@gatech.edu) [1], **Richard SAVERY** (rsavery3@gatech.edu) [1], and **Gil WEINBERG** (gilw@gatech.edu) [1]

[1] *Robotic Musicianship Lab*, **Georgia Institute of Technology**, Atlanta, GA USA

ABSTRACT

The task of classifying emotions within a musical track has received widespread attention within the Music Information Retrieval (MIR) community. Music emotion recognition has traditionally relied on the use of acoustic features, verbal features, and metadata-based filtering. The role of musical prosody remains under-explored despite several studies demonstrating a strong connection between prosody and emotion. In this study, we restrict the input of traditional machine learning algorithms to the features of musical prosody. Furthermore, our proposed approach builds upon the prior by classifying emotions under an expanded emotional taxonomy, using the Geneva Wheel of Emotion. We utilize a methodology for individual data collection from vocalists, and personal ground truth labeling by the artist themselves. We found that traditional machine learning algorithms when limited to the features of musical prosody (1) achieve high accuracies for a single singer, (2) maintain high accuracy when the dataset is expanded to multiple singers, and (3) achieve high accuracies when trained on a reduced subset of the total features.

1. INTRODUCTION

The work presented in this paper is situated in the intersection between research on emotion for robotics [1] and emotional classification research in Music Information Retrieval [2]. In particular, we focus on the under-explored domain of emotion-driven prosody for human-robot interaction [3]. Verbal prosody is concerned with elements of speech that are not individual phonetic segments but rather pertain to linguistic functions such as intonation, tone, stress, and rhythm. Similarly, musical prosody is defined as the performer's manipulation of music for certain expressive and coordinating functions [4]. It has been hypothesized that these expressive functions serve to communicate emotion [5].

In this paper, we explore the relationship between musical prosody and emotion through three research questions. First, are traditional machine learning algorithms able to accurately classify an individual's emotions when trained on only the features of musical prosody? Next, are these models able to generalize to a larger group of vocalists? Finally, which features of musical prosody contribute the most to the classification of emotion?

The paper is structured as follows, in Section 2, background and motivation are discussed. Section 3 describes the dataset collection, training and testing, the taxonomies used in classification, the feature extraction methodology and analysis of their relevance to emotion, feature aggregation, feature selection, and model generalization. Section 4 presents the experiments: Experiment 1 asks how well can traditional machine learning models classify emotion when limited to inputs of musical prosody, Experiment 2 explores our approach's ability to generalize to a larger population of singers, and Experiment 3 explores the individual contribution to accuracy of each feature via training on reduced subsets of the input vector. Section 5 provides discussion to these results, with particular attention paid to the relationships between emotions and potential future work. Finally, section 6 concludes the paper. A demo via python notebook with audio samples is available online. [1]

2. BACKGROUND

Emotion classification has been a major focus of research in recent years. Ekman created a discrete categorization that consists of fundamental basic emotions which are the root for more complex emotions [6]. Another classification model is the Circumplex model proposed by Posner et al which plots emotions on a continuous, two-dimensional scale of valence and arousal [7]. In this paper, we classify emotions using a model similar to the two-dimensional Circumplex model which is further described in section 3.1.

There has also been much work done in the field of analyzing emotion from text for tasks such as sentiment analysis. Research on classification of emotion in audio has taken many different approaches. Research into classifying emotions in knocking sounds has found that anger, happiness and sadness could be easily classified from audio alone [8]. There have been multimodal approaches which use audio in combination with another feature, namely visual facial features [9] [10] or text lyrics [11]. Furthermore, researchers have performed emotional classification from audio in the context of music by analyzing which musical features best convey emotions [12]. Panda et al. have found a relationship between melodic and

[1] https://github.com/brianmodel/EmotionClassification

dynamic features to a number of specific emotions [13]. Such features that were used to classify emotion in music, however, cannot be easily generalized to other domains. Prosody has been found by linguists to communicate emotion across various cultures, with patterns of pitch and loudness over time representing different emotions [14], and has shown the potential to improve human-robot interaction [15–17]. Our approach aims to bridge this gap by analyzing these prosodic features which are fundamental to everyday speech and explore how they can be used to classify emotional driven prosody.

Koo et al. have done work in speech emotion recognition using a combination of MFCC and prosodic features with a GRU model on the IEMOCAP dataset [18]. We expand upon their work by performing an in-depth analysis of 11 different audio features and their effect on classifying emotion. We also classify emotion beyond spoken language by analyzing prosodic features which better generalize to how humans convey emotion using the new dataset collected, as described in section 3.2.

3. METHODOLOGY

3.1 Taxonomy

One of the main challenges in emotional classification is the derivation of a taxonomy that accurately reflects the problem domain. The two common approaches to address this challenge are 1. Discrete emotional categorization; and 2. Continuous quantitative metrics of Valence and Arousal (sometimes called Control). We use both approaches with a categorical, as opposed to regression, approach to the latter.

Our models classify emotion under two taxonomies: first we categorize each data point as belonging to one of the twenty emotions located around the Geneva Wheel of Emotion. Then we categorize each data point as belonging to one of the quadrants depicted by the intersection of valance and control by assigning each emotion from the Geneva Wheel of Emotion to its respective quadrant. We abbreviate each of these quadrants as follows: "High Control Negative Valance": "HCN", "High Control Positive Valance": "HCP", "Low Control Negative Valance": "LCN", and "Low Control Positive Valance": "LCP". See Table 1 and Figure 1 for a visualization of the domain's taxonomy.

HCN	HCP	LCN	LCP
Anger	Amusement	Disappointment	Admiration
Contempt	Interest	Fear	Compassion
Disgust	Joy	Guilt	Contentment
Hate	Pleasure	Sadness	Love
Regret	Pride	Shame	Relief

Table 1. Selected emotional taxonomy for training

3.2 Data Collection

Due to a lack of data labeled with the appropriate taxonomy, we decided to collect and annotate a new dataset. To achieve this goal, we asked professional singers to consciously sing each emotion. To generate our dataset, three professional singers were tasked to improvise as many phrases as possible for each emotion in the Geneva Wheel of Emotion. The singers were instructed to sing each phrase between 1 and 20 seconds, and to spend approximately 15 minutes on each emotion, resulting in 4 to 6 hours of recordings per singer annotated with ground-truth labels.

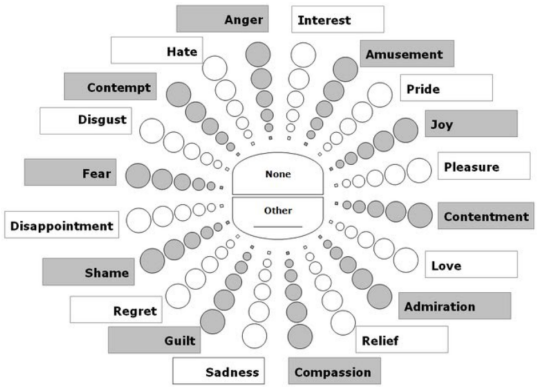


Figure 1. The Geneva Wheel of Emotion

Additionally, the singers were given the following instructions during their recording session:

1. Do not attempt to control for different intensities for each emotion
2. Sing anything for each phrase that you believe matches the emotion except use words.
3. After recording, mark any phrase that you believe did not capture the intended emotion and it will be deleted

3.3 Feature Extraction

In the following section, we define the features selected for extraction from our dataset prior to model training. Furthermore, we discuss each feature's relevance to emotional classification through an analysis of prior works.

3.3.1 Zero Crossing Rate

Zero Crossing Rate, the rate of sign-changes across a signal, is key in classifying percussive sounds. Unvoiced regions of audio are known to have higher Zero Crossing Rates [19]. One study analyzed ZCR for Anger, Fear, Neutral, and Happy signals and noted that higher peaks were found for Happy and Anger emotions [20].

3.3.2 Energy

Energy, the area under the squared magnitude of the considered signal, relates to the amount of spectral information in a signal [21] and previous studies have found energy is essential in distinguishing stressed and neutral speech [22].

3.3.3 *Entropy of Energy*

Entropy of Energy, the average level of "information" or "uncertainty" inherent within a signal's energy, has been shown in one study to have similar values for disgust and boredom [23]. To accurately measure the entropy of the different emotions, we must make sure we are not including parts of the signal where the individual is not speaking.

3.3.4 *Spectral Centroid*

Spectral Centroid, the power spectrum's center of mass, perceptually has a connection with a sound's brightness. It follows, that this parameter serves as an indicator of musical timbre [24]. Previous studies have shown spectral centroid is a significant component in music emotion [25].

3.3.5 *Spectral Spread*

Spectral Spread, the second central moment of the power spectrum, has shown to help the listener to differentiate noise-like and tone-like portions of a signal [26].

3.3.6 *Spectral Entropy*

Spectral Entropy, the entropy of the power spectrum, when used with MFCC features has shown an improvement in speech recognition accuracy [27]. Another study found spectral entropy to have the highest correlation to emotional valence of all features tested [28].

3.3.7 *Spectral Flux*

Spectral Flux, a measure of the rate of change of the power spectrum calculated as the Euclidean distance between sequential frames, relates to how fast the pitch changes in time and has been shown to be dominant in cross-domain emotion recognition from speech and sound and from sound and music [29].

3.3.8 *Spectral Rolloff*

Spectral Rolloff, the frequency under which some percentage of the total energy of the spectrum is contained, helps differentiate between harmonic content, characterized below the roll-off, and noisy sounds, characterized above the roll-off. Spectral rolloff has been shown to be one of the most important prosodic features in classifying emotion [28].

3.3.9 *MFCCs*

Mel-Frequency Cepstral Coefficients (MFCCs), a representation of the short-term power spectrum based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency, are used in speech recognition with their ability to represent the speech amplitude spectrum in a compact form [30]. Many studies have linked the importance of MFCC analysis to emotion recognition [20] [31] [32] .

3.3.10 *Chroma Vector and Deviation*

Chroma Vector, an approximation of the pitch class profiles present within a given frame and often used as the twelve tones, allows for the capture of harmonic and melodic characteristics while remaining robust toward changes in timbre and instrumentation. Previous studies have shown increases in emotional classification accuracy with chroma vector and its standard deviation [33, 34].

Parameter	Value
Mid-term Window Step	1.0 seconds
Mid-term Window Size	1.0 seconds
Short-term Window Step	0.05 seconds
Short-term Window Size	0.05 seconds

Table 2. Feature Aggregation Parameters

3.4 Feature Aggregation

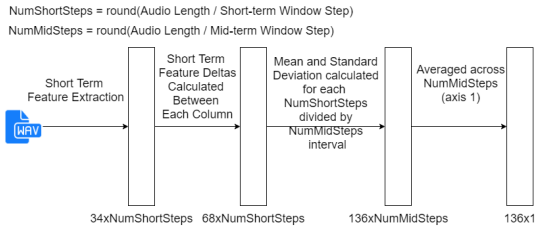


Figure 2. Model of Feature Aggregation

In this section, we define the aggregation pipeline from feature extraction to feature vector for each audio file. Figure 2 provides a visual modeling of our feature aggregation pipeline. Table 2 delineates the feature aggregation hyper parameters used in this study.

3.4.1 *Short-term Aggregation*

The short-term aggregation of a 5-second clip, using a Short-term Window Step of .05 seconds and a Short-term Window Size of .05 seconds is defined as follows: Each of the 34 features discussed above are extracted for every 50ms, resulting in 100 feature vectors of size 34x1, represented as a 34x100 matrix. Next, the deltas between each time step are calculated according to the equation $delta = feature\_vector - feature\_vector\_prev$. The first time stamp has all deltas set to 0. Each delta vector is concatenated onto its respective feature vector resulting in a size of 68x1, represented as a 68x100 matrix for the entire 5 second audio clip.

3.4.2 *Mid-term Aggregation*

Next, mid-term aggregation occurs with a Mid-term Window Size of 1.0 seconds and Mid-term Window Step of 1.0 seconds. The 68x100 matrix of Short-term features is split according to the ratio between the Mid-term and Short-term window size and step, resulting in 5 matrices of size 68x20. For each matrix, we calculate and flatten the mean and standard deviation for each row, resulting in 5 136x1 mid-term feature vectors, represented as a 136x5 matrix. Finally, we take the mean across the first axis resulting in a 136x1 feature vector representing our 5 second audio clip.

3.5 Classification

Prior work focused on musical classification has primarily found success in the implementation of k-nearest neighbor (K-NN) and support vector machines (SVM), finding the highest accuracies using SVMs [35]. In exploration of the relationship between musical prosody and emotion, we will implement a variety of machine learning models, namely we will train and evaluate KNNs, Linear SVMs, Random Forests, Extra Trees, Gradient Boosting, and Feed Forward Neural Networks (FFNN). FFNNs are used in experiment 3 only.

Experiment 1: we explore the base line accuracies, F-scores, and confusion matrices achieved by training each model with identical training, validation, and testing data from a single singer.

Experiment 2: we explore our model architecture's ability to generalize by expanding the dataset to include all 3 singers from data collection.

Experiment 3: we explore model performance on a reduced subset of the training feature, utilizing additive feature selection to compile a ranking of features.

4. RESULTS

4.1 Experiment 1

In experiment 1, we analyze the baseline accuracies, F-scores, and confusion matrices achieve by training KNNs, linear SVMs, Random Forests, Extra Trees, Gradient Boosting models on a single singer utilizing only the prosodic features outlined in the previous section. All models were trained with features extracted according to the parameters outlined in Table 2. Additionally, each model is optimized with respect to its associated hyper parameter. We optimize KNN for the number nearest neighbors, SVM for the soft margin, random forest for number of trees, gradient boosting for the number of boosting stages, and extra trees for the number of trees. [2]

Table 3 provides the best accuracy, F1-score, and selected hyper-parameter for each of our models trained on a Big 4 taxonomy for a single singer. All models perform better than twice the accuracy of random guessing, with the linear SVM and Gradient Boosting models achieving the highest accuracies. Further analysis of the confusion matrix of the Gradient Boosting model, shown in Figure 4, provides information about the classes that are most often confused for one another. The model struggles in distinguishing between Low Control Positive Valance and High Control Positive Valance. This is to say the model can tell that an individual is in a positive mood, but has difficulties distinguishing the Control or Arousal of the emotion.

Next, we examine classification under a single emotion taxonomy for a single singer. Table 4 shows the best accuracy, F1-score, and selected hyper-parameter for each of our models. Each model significantly outperforms random guessing. Even the worst model, the KNN, performs 6.5 times better than random chance (20 possible categories = 5% chance random guessing). Our best model, the linear SVM, performs approximately 10 times better than random guessing with an accuracy of 49.1%. The confusion matrix for the single emotion taxonomy has been included in Figure 3. Analysis of this confusion matrix yields a few observations: Disgust is rarely confused with other emotions, having the highest individual accuracy of 81.4%. Fear and Guilt are the two most common pair of emotions to be confused for one another. Pleasure is the most difficult emotion for the model to classify correctly, having the lowest individual accuracy of 18.6%.

Finally, our models perform extremely well when tasked with categorizing between two emotions, achieving accuracies as high as 98.9% with a f1 of 98.9 in the distinction between Love and Disgust using a SVM. This reinforces the intuition that by reducing the number of emotional categories we can achieve higher accuracies for identification.

Model	Accuracy	F1	Hyperparam
KNN	56.1	56.2	C=11
SVM	66.5	65.3	C=1.0
Extra Trees	64.6	64.3	C=100
Gradient Boosting	67.0	66.7	C=500
Random Forest	63.5	63.2	C=200

Table 3. Big 4 Taxonomy, 1 Singer Classification Results

Model	Accuracy	F1	Hyperparam
KNN	33.8	32.1	C=15
SVM	49.1	48.1	C=5.0
Extra Trees	44.3	42.8	C=500
Gradient Boosting	47.2	46.6	C=200
Random Forest	43.8	42.3	C=200

Table 4. Single Taxonomy, 1 Singer Classification Results

4.2 Experiment 2

Within machine learning, model generalization poses many challenges as models tend to memorize data and perform worse when exposed to new datasets. In experiment 2, we generalized our model by training on 3 different singers as opposed to training on one singer. Tables 5 and 6 compare the accuracies achieved by the various model architectures for 3 singers vs 1 singer.

With the exception of linear SVM, all model architectures maintain similar accuracies when trained on the 3 singer datasets. This maintenance of accuracy demonstrates the ability for traditional machine learning models to generalize well to a larger population when trained on only the features of musical prosody. We are unsure of why linear SVMs perform worse during generalization as compared to other models, seeing a drop of 6% in Big 4 taxonomy and a drop of 13% in single emotion taxonomy. This drop could potentially be a limitation in our methodology of only applying a linear kernel to SVM training, as perhaps an RBF or polynomial kernel would be better able to generalize to a larger population.

The results of this experiment are encouraging to the development of a general model of emotional classification based on musical prosody as accuracy is maintained when

[2] https://scikit-learn.org/

				HCN				HCP				LCN				LCP						
		Ang	Contem	Disg	Hate	Reg	Amu	Int	Joy	Ple	Pri	Disa	Fear	Gui	Sad	Sha	Adm	Com	Conten	Love	Rel	Total	Acc
	Ang	2.94	0.41	0.16	0.41	0	0.08	0	0.08	0.08	0.08	0	0.16	0.08	0	0.16	0	0	0.08	0.08	0.08	4.88	60.2%
	Contem	0.65	1.88	0.16	0.33	0.41	0.08	0.08	0.08	0.16	0	0.16	0.49	0.33	0	0.08	0.33	0	0	0	0.08	5.3	35.5%
HCN	Disg	0.24	0.08	5.31	0.08	0	0.33	0	0	0	0	0.16	0.16	0	0	0	0.08	0	0	0	0.08	6.52	81.4%
	Hate	0.57	0.33	0.24	3.35	0	0.16	0.08	0.33	0	0	0	0.16	0	0.16	0	0	0	0	0.24	0.08	5.7	58.8%
	Reg	0.24	0.08	0.08	0.16	1.47	0	0.08	0	0.41	0.16	0.16	0.08	0.49	0.08	0.16	0.08	0.41	0.16	0	0.16	4.46	33.0%
	Amu	0	0	0.16	0.08	0	3.18	0.33	0.33	0	0.08	0	0.08	0	0	0	0.16	0	0	0	0.08	4.48	71.0%
	Int	0	0.16	0	0	0.08	0.82	2.12	0	0.24	0.16	0.16	0	0	0.16	0.08	0.24	0.24	0.65	0.08	0.08	5.27	40.2%
HCP	Joy	0.08	0.16	0	0.33	0.08	0.57	0.08	2.53	0.41	0.08	0.08	0.08	0	0	0	0.49	0.16	0	0.33	0.24	5.7	44.4%
	Ple	0	0.16	0.08	0	0.24	0	0.41	0	1.06	0.16	0.24	0	0.41	0.08	0.24	0.41	0.73	0.49	0.82	0.16	5.69	18.6%
	Pri	0.16	0.08	0	0	0	0	0	0.08	0.08	2.37	0.08	0.08	0.08	0	0	0.24	0.16	0.16	0.33	0.16	4.06	58.4%
Truth	Disa	0.16	0	0.16	0	0	0	0.08	0.08	0.08	0.08	2.45	0.08	0.33	0	0.16	0.08	0.16	0	0	0.16	4.06	60.3%
	Fear	0.24	0.33	0.08	0.16	0.08	0	0	0	0.24	0.08	0.08	2.53	0.49	0.08	0.16	0.08	0.08	0	0.08	0.08	4.87	52.0%
LCN	Gui	0.08	0.08	0	0	0.24	0.08	0	0	0.24	0.24	0.08	1.14	2.45	0.08	0.16	0.08	0.08	0.24	0	0	5.27	46.5%
	Sad	0.33	0	0.16	0.24	0	0	0.24	0	0	0	0.24	0	0	1.71	0.16	0	0.08	0.16	0.08	0.24	3.64	47.0%
	Sha	0.08	0.16	0.08	0.16	0.33	0	0	0	0.08	0.33	0.24	0.65	0.33	0.08	1.55	0.08	0.16	0.33	0.16	0.08	4.88	31.8%
	Adm	0.08	0.08	0	0.16	0	0.41	0.24	0.24	0.16	0	0.24	0	0	0	0	2.29	0	0	0.08	0.08	4.06	56.4%
	Com	0	0.08	0	0	0	0	0.16	0	0	0.41	0.49	0.16	0.16	0	0.16	0.08	2.2	0.41	0.16	0	4.47	49.2%
LCP	Conten	0.08	0.08	0	0	0.16	0.16	0.08	0	0.33	0.08	0.08	0.08	0.16	0.24	0.08	0.49	0.98	1.71	0.08	0	4.87	35.1%
	Love	0.08	0.08	0	0.08	0	0.16	0.24	0.16	0.33	0.33	0	0.16	0.24	0.08	0	0.16	0.82	0.41	1.96	0	5.29	37.1%
	Rel	0.08	0.16	0.33	0.24	0	0	0.16	0.16	0.08	0	0.16	0	0.08	0.16	0	0.16	0.16	0.08	0.08	4	6.09	65.7%
	Total	6.09	4.39	7	5.78	3.09	6.03	4.38	4.07	3.98	4.64	5.1	6.09	5.63	2.91	3.15	5.53	6.42	4.88	4.56	5.84	100	49.1%

Figure 3. SVM, Individual Taxonomy, 1 Singers Confusion Matrix

		Prediction					
		HCN	HCP	LCN	LCP	Total	Acc
	HCN	20.82	2.38	2.46	1.39	27.05	77.0%
	HCP	2.21	15.16	2.38	5.66	25.41	59.7%
Truth	LCN	2.87	2.13	15.66	2.3	22.96	68.2%
	LCP	1.39	5.16	2.62	15.41	24.58	62.7%
	Total	27.29	24.83	23.12	24.76	100	67.0%

Figure 4. Gradient Boosting, Big 4 Taxonomy, 1 Singer Confusion Matrix

Model	1-S Accuracy	3-S Accuracy
KNN	56.1	57.9
SVM	66.5	60.6
Extra Trees	64.6	63.5
Gradient Boosting	67.0	68.8
Random Forest	63.5	65.1

Table 5. Big 4 Taxonomy, 1 Singer vs 3 Singer Accuracy

the dataset is expanded to a larger portion of the overall population.

4.3 Experiment 3

Experiment 3 analyzes model performance on a reduced subset of the feature vector for our single emotion taxonomy. Our implementation of Feature Selection follows an additive approach. We start with an empty permanent feature set and each feature is trained on its own. The feature with the highest f1 score is selected and added to our permanent feature set. This process is repeated until all features have been added to the permanent feature set. Finally, we plot the f1 score vs features used in model training.

Model	1-S Accuracy	3-S Accuracy
KNN	33.8	32.5
SVM	49.1	36.9
Extra Trees	44.3	42.7
Gradient Boosting	47.2	43.8
Random Forest	43.8	43.8

Table 6. Single Emotion Taxonomy, 1 Singer vs 3 Singer Accuracy

For 136 features, an additive feature selection training loop requires the training and f1 validation of 9316 models. Our initial training and validation was based on implementations using the python library sklearn. Unfortunately, sklearn does not provide native GPU training support and thus performing an additive feature selection using sklearn is not feasible with respect to training time. Our solution is to continue to use the feature selection and aggregation outlined above, and to replace the sklearn models with a Tensorflow feed forward neural net. All of these models look for statistical correlations between our features and the emotional classification. Thus the particular model should have minimal affect on the analysis of feature importance performed by additive feature selection. Training was done sequentially on a RTX 3090 using CUDA v11 and took just under 24 hours to train and validate all 9316 models.

Our feed forward neural net contained the input layer, two dense layers of 136 nodes with relu activation functions, and a dense 20 node output layer. We trained using a Sparse Categorical Cross entropy loss function optimized using an Adam optimizer with 5 epochs per model.

Figure 7 shows the F1 score achieved vs the Feature included in the model pipeline. All feature on and to the right of any point in the x axis are included in training. An F1 of 45 is achieved within the first 25 features. Furthermore, the addition of the remaining 111 features only increases our F1 score to 52. This graph emphasizes the importance of spectral roll-off and MFCC 7 in the classification of emotion, as aggregations of these two features allow for an F1 score just below 20 with 4 total features.

5. DISCUSSION

5.1 Analysis

We demonstrate that prosodic features can be used to classify human emotions, achieving high accuracies on classifying emotions for a single singer dataset as seen in tables 3 and 4. Furthermore, we obtained encouraging results

		Prediction																					
				HCN					HCP			LCN						LCP					
		Ang	Contem	Disg	Hate	Reg	Amu	Int	Joy	Ple	Pri	Disa	Fear	Gui	Sad	Sha	Adm	Com	Conten	Love	Rel	Total	Acc
	Ang	1.74	0.34	0.59	0.5	0.07	0.32	0.14	0.07	0.02	0.34	0.09	0.02	0.09	0.02	0.07	0.07	0.05	0	0.02	0.09	4.65	37.4%
	Contem	0.27	2.06	0.25	0.41	0.25	0.09	0.18	0.25	0.18	0.02	0.2	0.2	0.11	0.09	0.25	0.2	0.07	0.02	0.05	0.27	5.42	38.0%
HCN	Disg	0.45	0.5	2.58	0.68	0.02	0.45	0.07	0.07	0.02	0.16	0.02	0.05	0.07	0.02	0.05	0.11	0	0.02	0	0.2	5.54	46.6%
	Hate	0.38	0.38	0.57	2.47	0.05	0.11	0.07	0.29	0.16	0.27	0.09	0.05	0.02	0.05	0.16	0.07	0.05	0.09	0.14	0.2	5.67	43.6%
	Reg	0.02	0.34	0.02	0.11	2.26	0	0.07	0	0.23	0.07	0.07	0.18	0.18	0.14	0.11	0.11	0.34	0.05	0.23	0.34	4.87	46.4%
	Amu	0.09	0.02	0.34	0.11	0	2.38	0.29	0.38	0.2	0.45	0	0.02	0	0	0	0.18	0.05	0.25	0.05	0.16	4.97	47.9%
	Int	0	0.14	0.11	0.05	0.07	0.2	2.56	0.34	0.05	0.05	0.09	0.29	0.14	0.25	0.25	0.02	0.07	0.14	0.07	0.11	5	51.2%
HCP	Joy	0.07	0.14	0.07	0.16	0.09	0.45	0.11	2.87	0.09	0.07	0.05	0.23	0.07	0.18	0.07	0.14	0.05	0	0	0.09	5	57.4%
	Ple	0.05	0.16	0.07	0.16	0.07	0.16	0.02	0.07	1.92	0.45	0.11	0.07	0.02	0	0.11	0.29	0.25	0.75	0.27	0.2	5.2	36.9%
	Pri	0.09	0.09	0.05	0.16	0.07	0.34	0.14	0.27	0.38	2.35	0	0.14	0	0.02	0.18	0.23	0.07	0.34	0.16	0.36	5.44	43.2%
Truth	Disa	0.14	0.11	0.07	0.09	0.11	0	0.11	0	0.25	0.09	1.83	0.25	0.25	0.2	0.25	0.05	0.2	0.07	0.07	0.16	4.3	42.6%
	Fear	0.09	0.27	0.05	0.09	0.07	0.05	0.25	0.11	0.23	0.05	0.16	2.85	0.27	0.18	0.29	0.05	0.11	0	0.11	0.05	5.33	53.5%
LCN	Gui	0.07	0.11	0.07	0.02	0.09	0	0.11	0.07	0.14	0.02	0.32	0.54	1.67	0.07	0.23	0.09	0.11	0.02	0.14	0.07	3.96	42.2%
	Sad	0	0.05	0.02	0.09	0.09	0	0.2	0.09	0.11	0.02	0.07	0.48	0.14	2.53	0.29	0.02	0.16	0.05	0.11	0.11	4.63	54.6%
	Sha	0.14	0.16	0.14	0.14	0.14	0	0.16	0.07	0.07	0.07	0.25	0.45	0.25	0.34	1.97	0.05	0.07	0.16	0.14	0.14	4.91	40.1%
	Adm	0.07	0.11	0.07	0.23	0.27	0.18	0.07	0.18	0.36	0.25	0.09	0.07	0.02	0.18	0.09	1.24	0.27	0.11	0.11	0.32	4.29	28.9%
	Com	0	0.2	0.09	0.14	0.2	0	0.02	0	0.27	0.14	0.05	0.07	0.07	0.07	0.11	0.14	1.76	0.34	0.5	0.36	4.53	38.9%
LCP	Conten	0	0.09	0.02	0.11	0.11	0.11	0.07	0.07	0.61	0.36	0.14	0	0.05	0.02	0.14	0.16	0.32	1.74	0.34	0.41	4.87	35.7%
	Love	0	0.07	0	0.14	0.2	0.05	0.07	0.11	0.57	0.16	0.14	0.11	0.05	0.05	0.02	0.41	0.48	0.38	1.61	0.5	5.12	31.4%
	Rel	0.11	0.07	0.29	0.2	0.32	0.16	0.14	0.07	0.16	0.23	0.09	0	0	0.07	0	0.29	0.18	0.23	0.5	3.35	6.46	51.9%
	Total	3.78	5.41	5.47	6.06	4.55	5.05	4.85	5.38	6.02	5.62	3.86	6.07	3.47	4.48	4.64	3.92	4.66	4.76	4.62	7.49	100	43.8%

Figure 5. Gradient Boosting, Individual Taxonomy, 3 Singers Confusion Matrix

		Prediction					
		HCN	HCP	LCN	LCP	Total	Acc
	HCN	17.81	2.73	2.51	3.05	26.1	68.2%
	HCP	2.42	16.45	2.44	4.23	25.54	64.4%
Truth	LCN	2.08	1.76	17.4	1.81	23.05	75.5%
	LCP	2.46	3.98	1.74	17.13	25.31	67.7%
	Total	24.77	24.92	24.09	26.22	100	68.8%

Figure 6. Gradient Boosting, Big 4 Taxonomy, 3 Singer Confusion Matrix

regarding the model's generalization between singers as demonstrated by tables 5 and 6. However, given our limited dataset, more research is needed to study how the models generalize for additional singers with different voices.

Our feature selection aligns with prior research indicating that energy and MFCC were the most useful features for classifying emotion [9]. However, we have been able to show that the results holds true not just for phonological speech, but in the more specific domain of musical prosody.

5.2 Relationships between Emotions

The classification results give us new insights into the uniqueness and relationships between emotions. Looking at the individual classification data between all the singers in Figure 3, we can see how the model was best able to classify fear, joy and relief. This is in contrast to emotions such pleasure or admiration which showed the lowest classification accuracy. These results demonstrate the manner in which different humans convey emotions, and what emotions are similarly expressed by different individuals. When conveying relief, all three singers expressed a diminuendo and exhale. Similarly, when conveying fear all three singers expressed a crescendo and more accented tones. On the other hand, there was a high level of variation when conveying pleasure, with many different tone ranges, mouth shapes, etc. being present in the data.

Furthermore, from the confusion matrix in Figure 5, we can see that the emotion pairs of Hate and Disgust as well as Pleasure and Contentment are the most common emotions to be misclassified as one another. We suggest that this is due to these emotions representing similar meanings, thus they would be conveyed using similar features. For instance, Hate and Disgust both tend to consist of lower tones while Pleasure and Contentment have higher tones.

5.3 Future Work

One of the major challenges we faced was the limited amount of data that was collected. We plan on expanding this dataset to a larger variety of singers and other instrumentalists so that we can better understand how the models can generalize to different sounds. Additional future work includes developing a more sophisticated deep-learning based model on the raw audio data for classifying emotion using the expanded dataset we will collect. This will allow the model to make predictions beyond what could be possible using the features we chose in our feature selection. It would open up the potential to achieve much higher accuracy and better model generalization.

6. CONCLUSIONS

Our novel dataset using an expanded emotion taxonomy provides opportunity for the development of a more articulate understanding of emotions. Previous attempts to correlate emotion to audio or music are based on fewer emotions, and often rely on lyrics or song metadata for classification. Our algorithms demonstrate a high level of accuracy on a 20 category taxonomy for emotions, utilizing only prosodic features. By restricting the type of input data to prosodic features and expanding the number of classified emotions, our models can be used for a wide range of research challenges within the domain of emotional classification. Furthermore, we have demonstrated that our approach is able to generalize to a larger subset of the overall population. Finally, the restriction of our feature vector via additive feature selection demonstrates the ability for prosodic features to achieve a high-level accuracy for emotional classification for a relatively small number of features.

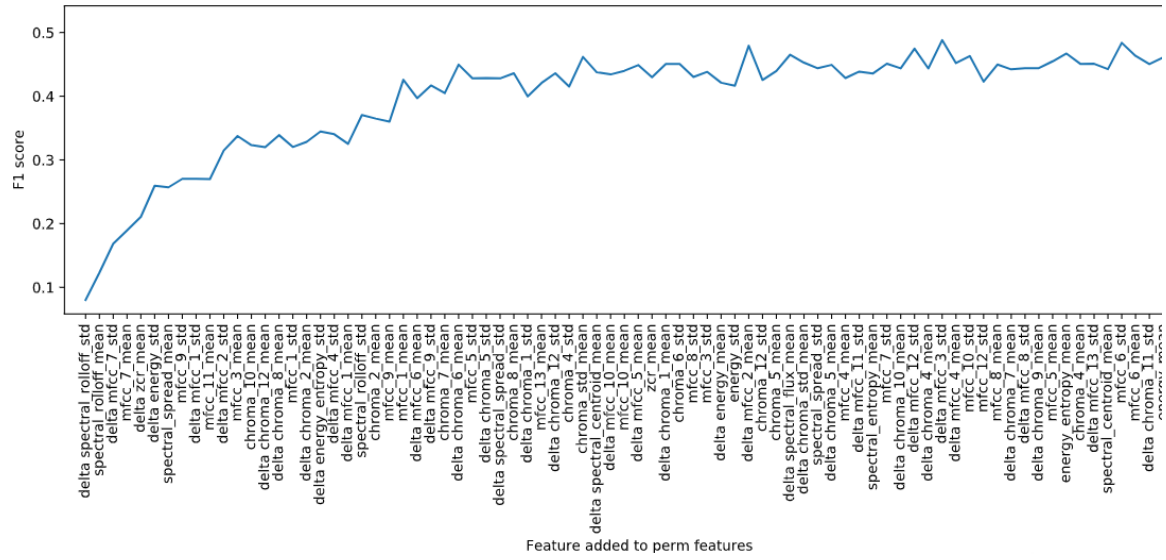


Figure 7. F1 score vs Features included in model pipeline

7. REFERENCES

[1] R. Savery and G. Weinberg, "A survey of robotics and emotion: Classifications and models of emotional interaction," 07 2020.

[2] P. Cano, M. Koppenberger, and N. Wack, "Content-based music audio recommendation," in *Proceedings of the 13th annual ACM international conference on Multimedia*, 2005, pp. 211–212.

[3] R. Savery, R. Rose, and G. Weinberg, "Establishing human-robot trust through music-driven robotic emotion prosody and gesture," in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2019, pp. 1–7.

[4] C. Palmer and S. Hutchins, "What is musical prosody?" *Psychology of Learning and Motivation*, vol. 46, 12 2006.

[5] P. N. Juslin and J. A. Sloboda, *Music and emotion: Theory and research.* Oxford University Press, 2001.

[6] P. Ekman, "Basic emotions," *Handbook of cognition and emotion*, vol. 98, no. 45-60, p. 16, 1999.

[7] J. Posner, J. A. Russell, and B. S. Peterson, "The circumplex model of affect: an integrative approach to affective neuroscience, cognitive development, and psychopathology," *Development and psychopathology*, vol. 17, no. 3, pp. 715–734, 2005.

[8] M. Houel, A. Arun, A. Berg, A. Iop, A. Barahona-Rios, and S. Pauletto, "Perception of emotions in knocking sounds : an evaluation study," in *:*, 2020, qC 20200722. [Online]. Available: https://smc2020torino.it/adminupload/file/SMCCIM_2020_paper_95.pdf

[9] S. ul haq, P. Jackson, and J. Edge, "Audio-visual feature selection and reduction for emotion classification," *The proceedings of international conference on auditory-visual speech processing*, pp. 185–190, 09 2008.

[10] L. C. De Silva and Pei Chi Ng, "Bimodal emotion recognition," in *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, 2000, pp. 332–335.

[11] A. Jamdar, J. Abraham, K. Khanna, and R. Dubey, "Emotion analysis of songs based on lyrical and audio features," *CoRR*, vol. abs/1506.05012, 2015. [Online]. Available: http://arxiv.org/abs/1506.05012

[12] Y. Song, S. Dixon, and M. Pearce, "Evaluation of musical features for emotion classification," 10 2012.

[13] R. Panda, R. M. Malheiro, and R. P. Paiva, "Audio features for music emotion recognition: a survey," *IEEE Transactions on Affective Computing*, pp. 1–1, 2020.

[14] R. W. Frick, "Communicating emotion: The role of prosodic features." *Psychological Bulletin*, vol. 97, no. 3, p. 412–429, 1985.

[15] R. Savery, R. Rose, and G. Weinberg, "Finding shimi's voice: fostering human-robot communication with music and a nvidia jetson tx2," in *Proceedings of the 17th Linux Audio Conference*, 2019, p. 5.

[16] R. Savery, L. Zahray, and G. Weinberg, "Emotional musical prosody for the enhancement of trust in robotic arm communication," in *Trust, Acceptance and Social Cues in Human-Robot Interaction: 29th IEEE International Conference on Robot & Human Interactive Communication*, 2020.

[17] ——, "Before, between, and after: Enriching robot communication surrounding collaborative creative activities," *Frontiers in Robotics and AI*, vol. 8, p. 116, 2021. [Online]. Available: https://www.frontiersin.org/article/10.3389/frobt.2021.662355

[18] H. Koo, S. Jeong, S. Yoon, and W. Kim, "Development of speech emotion recognition algorithm using mfcc and prosody," in *2020 International Conference on Electronics, Information, and Communication (ICEIC)*, 2020, pp. 1–4.

[19] Wai Pang Ng, J. M. H. Elmirghani, R. A. Cryan, Yoong Choon Chang, and S. Broom, "Divergence detection in a speech-excited in-service non-intrusive measurement device," in *2000 IEEE International Conference on Communications. ICC 2000. Global Convergence Through Communications. Conference Record*, vol. 2, 2000, pp. 944–948 vol.2.

[20] E. Ramdinmawii, A. Mohanta, and V. K. Mittal, "Emotion recognition from speech signal," in *TENCON 2017 - 2017 IEEE Region 10 Conference*, 2017, pp. 1562–1567.

[21] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the cuidado project," *CUIDADO Ist Project Report*, vol. 54, no. 0, pp. 1–25, 2004.

[22] O.-W. Kwon, K. Chan, J. Hao, and T.-W. Lee, "Emotion recognition by speech signals," in *Eighth European Conference on Speech Communication and Technology*, 2003.

[23] S. Lalitha, A. Mudupu, B. V. Nandyala, and R. Munagala, "Speech emotion recognition using dwt," in *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*. IEEE, 2015, pp. 1–4.

[24] R. Kendall and E. Carterette, "Difference thresholds for timbre related to spectral centroid," in *Proceedings of the 4th International Conference on Music Perception and Cognition, Montreal, Canada*, 1996, pp. 91–95.

[25] B. Wu, A. Horner, and C. Lee, "Musical timbre and emotion: The identification of salient timbral features in sustained musical instrument tones equalized in attack time and spectral centroid," in *ICMC*, 2014.

[26] U. Jain, K. Nathani, N. Ruban, A. N. J. Raj, Z. Zhuang, and V. G. Mahesh, "Cubic svm classifier based feature extraction and emotion detection from speech signals," in *2018 International Conference on Sensor Networks and Signal Processing (SNSP)*. IEEE, 2018, pp. 386–391.

[27] A. Toh, R. Togneri, and S. Nordholm, "Spectral entropy as speech features for speech recognition," *Proceedings of PEECS*, 01 2005.

[28] S. Cunningham, J. Weinel, and R. Picking, "High-level analysis of audio features for identifying emotional valence in human singing," in *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*, ser. AM'18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3243274.3243313

[29] F. Weninger, F. Eyben, B. W. Schuller, M. Mortillaro, and K. R. Scherer, "On the acoustics of emotion in audio: what speech, music, and sound have in common," *Frontiers in psychology*, vol. 4, p. 292, 2013.

[30] B. Logan *et al.*, "Mel frequency cepstral coefficients for music modeling." in *Ismir*, vol. 270. Citeseer, 2000, pp. 1–11.

[31] K. V. Krishna Kishore and P. Krishna Satish, "Emotion recognition in speech using mfcc and wavelet features," in *2013 3rd IEEE International Advance Computing Conference (IACC)*, 2013, pp. 842–847.

[32] D. Neiberg, K. Elenius, and K. Laskowski, "Emotion recognition in spontaneous speech using gmms," in *Ninth international conference on spoken language processing*, 2006.

[33] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbull, "Music emotion recognition: A state of the art review." 2010, pp. 255–266.

[34] E. M. Schmidt and Y. E. Kim, "Learning emotion-based acoustic features with deep belief networks," in *2011 IEEE workshop on applications of signal processing to audio and acoustics (Waspaa)*. IEEE, 2011, pp. 65–68.

[35] K. Bischoff, S. Claudiu, R. Paiu, W. Nejdl, C. Laurier, and M. Sordo, "Music mood and theme classification - a hybrid approach." 01 2009, pp. 657–662.

PERCEPTION OF EMOTIONS IN MULTIMODAL STIMULI: THE CASE OF KNOCKING ON A DOOR

Alessandro IOP (aiop@kth.se) [1] and **Sandra PAULETTO** (pauletto@kth.se) [1]

[1] **KTH Royal Institute of Technology**, Stockholm, Sweden

ABSTRACT

Knocking sounds are highly expressive. In our previous research we have shown that from the sound of knocking actions alone a person can differentiate between different basic emotional states. In media productions, such as film and games, knocks can be very important storytelling devices as they allow the story to transition from one part to another. Research has shown that colours can affect our perception of emotions. However the relationship between colours and emotions is complex and dependent on multiple factors. In this study we investigate how the visual characteristics of a door, more specifically its colour, texture and material, presented together with emotionally expressive knocking actions, can affect the perception of the overall emotion evoked in the audience. Results show that the door´s visual characteristics have little effect on the overall perception of emotions, which remains dominated by the emotions expressed by the knocking sounds.

1. INTRODUCTION

Knocking on a door is a very common everyday sound. Much information can be conveyed through such a simple yet expressive action: from perceiving the way the knock is performed (e.g. closed or open palm), to recognising the emotional intention of the person knocking on the door. Understanding how communication through everyday sounds takes place, and in particular how emotions can be recognised through these sounds, is of fundamental importance when designing and synthesising everyday sounds with a specific intention to be conveyed. Media industries such as gaming, advertising and cinema can benefit from technologies informed by knowledge of cross-modal perception in order to produce the desired effects on their audiences. Research on how emotional intentions are expressed in everyday sounds is relatively recent, especially in comparison to what we know about emotions and music or voice. A number of studies in recent years have expanded on the knowledge of human perception of emotions in aural stimuli of different nature [1–4]. Within this broader field of research, there is little exploration of the effect of knocking sounds on the emotions perceived by a listener. The aim of the present study is to build upon previous research on everyday sounds and emotions by focusing on audiovisual integration in multimodal stimuli of knocking sounds. More specifically, we assess the effect that audiovisual integration has on the perception of five basic emotional states (anger, fear, happiness, sadness and neutral). We investigate how audio and visual modalities, carrying congruent as well as non-congruent emotional information, interact in a simple representation of a knocking action performed on a door, and contribute to producing the perception of an overall emotion. In this regard, Gerdes et al. [5] explored how audio and visual cues interact to steer attention. The study shows that emotional auditory cues guide visual spatial allocation of attention specifically to emotionally congruent pictures.

We conducted a pre-study and an audiovisual experiment. The pre-study involved visual-only stimuli of doors of different colours, materials and textures. The aim of this study was to use its results, in conjunction with findings from literature on colour and emotions, to select 5 different coloured doors associated with anger, fear, happiness, sadness and neutral. In the subsequent audiovisual experiment, we combined 5 knocking action sounds, which in our previous study [6] were rated to be strongly associated with the same 5 basic emotions, with the 5 coloured doors. In this experiment, we aimed to investigate whether the combination of the appearance of the door with an emotional knocking sound could affect the overall emotional perception [1].

The next sections are organised as follows: in Background (§2) the most relevant previous research and theoretical background is reviewed; in Method (§3) the description of the experimental design is presented; in Results (§4) a summary of the statistical analysis of the collected data is reported; in Discussion (§5) results are discussed in light of previous research, and in Conclusions (§6) the work is summarised and directions for further work are outlined.

2. BACKGROUND

Everyday sounds can communicate complex information [7]. Furthermore, even sounds without explicit connection with everyday objects or actions, such as tones and noise complexes, can produce an emotional reaction [8]. Furthermore, emotions, as well as other characteristics such as material and shape of an object [3, 9, 10], are an integral part of auditory perception and are used to categorise ev-

[1] see stimuli from pre-study and audiovisual experiment here: https://kth.box.com/s/ske2j9gzzl7eclnzehlrlfzqt7hy3acq

eryday sounds [4]. From someone´s footstep, for example, we can infer many characteristics of the walker including gender, type of sole, and emotional intentions [2]. In regard to knocking sounds, our recent research has shown that basic emotional intentions such as anger, fear, happiness, sadness, and neutral state can be recognised from listening to knocking sounds alone [11]. Additionally, when utilising a large dataset of knocking action sounds produced by a professional Foley artist the degree of emotion recognition increases, showing only confusion between the labelling of anger and fear [6]. We also showed that emotion-specific acoustic patterns in knocking sounds confirm findings from previous research in speech and music performance [2, 12]

In this study, we selected 5 knocking sound actions from our data-set [2] of professionally performed knocking actions that were most strongly associated with anger, fear, happiness, sadness, and neutral state. We then combined them with visual representations of doors, which were created using Blender [3]. The design of these images was informed by research on colour, material and texture (i.e. the roughness and pattern of the surface). Research shows that colours can affect our emotional perception [13–15]. What emerges is a general agreement between most authors on a few colours (e.g. blue, red and yellow), although there is little consistency in the framework adopted for defining and categorising the colours and the emotions associated to them. Additionally, research shows that associations between colours and emotions depend on cultural factors [16, 17] as well as other aspects such as age [18–21]. Despite this complex picture, practical knowledge about colours is applied in many areas such as media production [22], marketing [23] or interior design [24]. Research on the association between materials and emotions, or textures and emotions appears to be limited. Crippa et al. [25] found that different materials can evoke emotions, even if weakly, such as satisfaction, joy, fascination, dissatisfaction and boredom. In relation to texture and emotion, Ebe and Umemuro [26] and Iosifyan and Korolkova [27] have found that people significantly associate basic emotions to different textures perceived through touch. For this study, we created 32 doors with different colours, materials and textures. Then, by combining the results of our the pre-study with findings from research on colour and emotion, we selected 5 doors that are strongly associated with the 5 basic emotions explored in this work.

3. METHOD

3.1 Pre-study

3.1.1 Stimuli

Thirty two images of closed doors (600 × 600 px), combining 8 colours (yellow, blue, black, grey, green, white, red, brown) and 4 materials plus textures (metal, smooth wood, intermediate wood, rough wood), were rendered using Blender 2.90.0 in a neutral indoor environment comprised of an off-white surrounding wall, a light grey floor and basic door features (door frame of the same material of the door and a simple metallic-grey handle (e.g. Figure 1)). Six out of eight colours were chosen from the most frequently studied in previous colour and emotion research, while grey and brown were chosen as being the colours most commonly associated to a door of the selected materials.

Figure 1. Example of a door image used in the pre-study and the audiovisual experiment, depicting a red door with an intermediate wood texture.

3.1.2 Procedure

An online survey was created using the online platform PsyToolkit [4]. Participants were presented with the 32 images of doors. For each image, participants were asked to choose which emotional state the door evoked. The order of the stimuli and the options available for each question were randomised. Finally, for each door, participants were asked to select the colour and material/texture of the door by answering two separate single-choice questions. This allowed researchers, who did not have control over the viewing monitor, to confirm that participants viewed the visual characteristics of the doors correctly.

3.1.3 Participants

Twenty four participants participated in the survey. Six did not complete the survey and their results were therefore excluded. The remaining 18 participants (11 female, 7 male) were aged 19-65 (11 between 19 and 25; 5 between 26 and 35; 1 between 36 and 50; 1 above 50). None of the participants were colourblind.

3.2 Audiovisual experiment

3.2.1 Stimuli

In this experiment we used 30 audiovisual stimuli. These combined the 6 most strongly emotionally expressive knocking action sounds from our dataset recorded by a

[2] http://doi.org/10.5281/zenodo.3668503
[3] https:https://www.blender.org
[4] https://www.psytoolkit.org

professional Foley artist, one per each of the 5 basic emotions considered plus one additional neutral knocking action, and 5 images of doors associated with the same basic emotions selected from the pre-study stimuli. The reason for having two neutral knocking sounds was to be able to investigate in more detail whether the lack of a strongly recognisable emotion in the sound (neutral state) would allow for the emotion evoked by the visuals to affect the overall emotional perception more strongly. Finally, in order to select the 5 doors we combined the results from the pre-study (see §4) with findings from the literature in colour and emotions. Here the main selection criteria:

1. the door must be among those significantly associated, in the pre-study, with the emotion in question;
2. the door must have a relatively high absolute number of responses in the pre-study for the emotion considered;
3. associations between door characteristics and emotion must be confirmed, wherever possible, by previous research. [13, 16, 28] specifically support the red-anger pair, [20, 28] explicitly support the blue-happiness pair, [14, 29] associate yellow to negative valence/unpleasant feelings, and [14, 21] associate grey to negative valence and low arousal.

The doors selected were: anger = red + intermediate wood (RIW); fear = yellow + rough wood (YRW); happiness = blue + metal (BM); neutral = grey + smooth wood (GSW); sadness = brown + metal (BrwM).

3.2.2 Procedure

An online survey was created in PsyToolkit. Participants who had participated in the pre-study, or were knowledgeable about colour or sound theory were excluded. Before starting the test, participants were asked to adjust the volume in their headphones in order to be able to comfortably perceive both the softest and the loudest sounds used in the experiment. For each stimulus, the evoked emotional state was tested as a single-choice question. The order of the stimuli and the options available for each question were randomised.

3.2.3 Participants

One hundred and seven participants participated in the experiment. Among these, 52 did not complete the survey or had participated to our previous pre-study, and 20 had knowledge about colour or sound theory, and were therefore excluded from the analysis. Of the remaining 35 (15 female, 20 male) none were colourblind and their ages ranged as follows: 3 between 18 or below, 10 between 19 and 25, 5 between 26 and 35, 3 between 36 and 50, 14 between 51 and 65.

4. RESULTS

4.1 Pre-study

The Chi-Square test shows that there is a significant relationship between the doors´ colour and the perceived emotions $\chi^2(28, N = 576) = 110.313$, $p < .01$. When considering the z-test pair-wise comparisons with a Bonferroni correction, we find that each emotion is significantly associated with a number of colours.

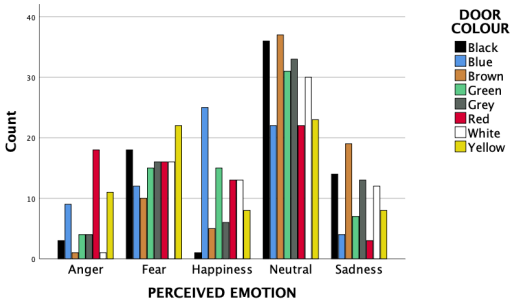


Figure 2. Pre-study: Door's colour vs perceived emotion.

Anger is significantly associated with red, yellow, blue. Red rough wood and red intermediate wood have the highest number of responses. Happiness is significantly associated with blue, green, red, white. Blue metal and blue intermediate wood have the highest number of responses. Sadness is significantly associated with brown, black, grey, green, white, yellow. Brown metal has the highest number of responses for sadness. In regard to fear, there is no significant difference in association with any of the colours. Yellow rough wood has the highest number of responses for fear. Similarly, for neutral state, there is no significant difference in association with any of the colours. Grey Smooth wood has the highest number of responses for neutral. When focusing on emotions and material/texture we find that fear is significantly associated with rough wood, while happiness and neutral are not associated with rough wood. Finally, the colours were correctly recognised (94.4% of the time, on average), with the sole exception of the metal white door, which was confused with the metal grey door 38.8% of the time. The material of the door, metal or wood, was recognised correctly. The intermediate wood texture was at times confused with smooth wood or rough wood (53.5% of the time). Overall, by combining these results and findings from previous research (see §3.2.1), the following doors were selected to be utilised in the audiovisual experiment: RIW for anger, YRW for fear, BM for happiness, GSW for neutral, BrwM for sadness.

4.2 Audiovisual experiment

We found a significant relationship between the intended emotion of the knocking action sound and the perceived overall emotion $\chi^2(16, N = 1050) = 803.651$, $p < .01$. There is no statistical significant difference between results for anger and fear. This confirms results from our previous study [6] from which the knocking actions where selected. Happiness, sadness and neutral are recognised correctly with statistical significance (see Figure 3).

There is no significant relationship between the doors' intended emotions and the perceived overall emotions $\chi^2(16,$

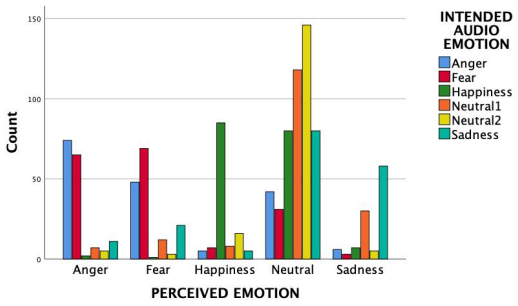


Figure 3. Audiovisual experiment: Perceived emotion vs intended audio emotion.

N = 1050) = 20.666, p > .05. The visual characteristics of the doors do not contribute to the overall emotional perception of these audiovisual stimuli. The sound of the knocking action, on this occasion, dominates the overall perceived emotion. In regard to congruent audiovisual stimuli (i.e. where the intended emotions for audio and visual aspects coincide), they do not seem to be significantly more expressive, however we note that the RIW door combined with the angry knock has the highest number of responses for perceived anger; the BrwM door combined with the sad knock reports the second highest number of responses for perceived sadness (GS with sad knock reports the highest); the BM door combined with the happy knock reports the second highest number of responses for perceived happiness together with BrwM with happy knock (RIW with happy sound has the highest number of responses). Finally, the intended emotions of the doors such as anger, fear, happiness and sadness do not come through more strongly in the overall perceived emotion when the knocking sound´s intended emotion is neutral.

5. DISCUSSION

Results from the pre-study are consistent with trends found in previous studies on the association between colours and emotions. More specifically, high-arousal emotions are often matched with warmer colors like red and yellow [19, 20, 22, 23], or highly saturated colours [14, 19, 28]. Our results also confirm the complexity of the area. While associations between single emotions and a few colours were detected, no one-to-one associations between emotional states and colours. We suggest that if one-to-one colour-emotions associations exist, they might require a larger sample size to detect them. The most important result from the audiovisual experiment is the dominance of the aural modality on the perception of the overall emotion. It appears that, in this case, the audio drives the emotional state evoked in the subjects, a conclusion which, we speculate, could be due to the different implied sources of the audio and visuals respectively. While a knocking sound would usually imply the presence of a human as its source (i.e. an agent experiencing emotions), the colour and material of a door are features of an inanimate object that does not experience emotions (although we might project our emotions onto this object). We therefore suggest that, in this case, the emotion behind an human action, here portrayed by the audio modality, bears more importance than the emotion evoked by the door. This might tell us that, in the context of filmmaking, for example, Foley performances contribute more than we think to the overall storytelling experience, and should perhaps be considered with the same attention traditionally granted to other visual aspects of the process (e.g. production design). In regard to congruent and not congruent stimuli, we did not find significant results. However looking at absolute number of responses for some emotions congruence suggests an increase in effect. With a larger study the presence of such small effects might be confirmed.

6. CONCLUSIONS

With the goal of exploring the perception of emotion in audiovisual representations of everyday actions, and building on previous research in the field, we have conducted two experiments. These enabled us to investigate how basic emotions can be evoked through a combinations of audio and visual features, and assess the impact that the different modalities have on the perceived emotions. Overall, the results provide a basis to help us understand how different aspects of an audiovisual artifact might contribute to the formation of an overall emotional experience in an audience. Further work could include investigating the impact of several other features on the perception of emotions. For example, point-light animations of the knocking hand could clarify the point of view and point of audition, door size, and audio and visual contextual elements could be added to the investigation. Furthermore, a wider range of emotions could be investigated, perhaps using different frameworks such as arousal-valence, as well as the impact of different immersive playback environments (cinematic, VR, gaming environment, etc.) on the perception of emotions in audiovisual stimuli.

7. REFERENCES

[1] X. Li, R. J. Logan, and R. E. Pastore, "Perception of acoustic source characteristics: Walking sounds," *The Journal of the Acoustical Society of America*, vol. 90, no. 6, pp. 3036–3049, 1991.

[2] B. L. Giordano, H. Egermann, and R. Bresin, "The production and perception of emotionally expressive walking sounds: Similarities between musical performance and everyday motor activity," *PLoS One*, vol. 9, no. 12, p. e115587, 2014.

[3] T. L. Bonebright, "Perceptual structure of everyday sounds: A multidimensional scaling approach." Georgia Institute of Technology, 2001.

[4] P. Bergman, D. Västfjäll, A. Tajadura-Jiménez, and E. Asutay, "Auditory-induced emotion mediates perceptual categorization of everyday sounds," *Frontiers in psychology*, vol. 7, p. 1565, 2016.

[5] A. Gerdes, M. J. Wieser, and G. W. Alpers, "Emotional pictures and sounds: a review of multimodal interactions of emotion cues in multiple domains," *Frontiers in Psychology*, vol. 5, p. 1351, 2014.

[6] A. Barahona-Rıos and S. Pauletto, "Synthesising knocking sound effects using conditional wave-gan," in *17th Sound and Music Computing Conference, Online*, 2020.

[7] M. Marcell, M. Malatanos, C. Leahy, and C. Comeaux, "Identifying, rating, and remembering environmental sound events," *Behavior research methods*, vol. 39, no. 3, pp. 561–569, 2007.

[8] D. Västfjäll, "Emotional reactions to sounds without meaning," *Psychology*, vol. 3, no. 8, p. 606, 2012.

[9] S. McAdams, A. Chaigne, and V. Roussarie, "The psychomechanics of simulated sound sources: Material properties of impacted bars," *The Journal of the Acoustical Society of America*, vol. 115, no. 3, pp. 1306–1320, 2004.

[10] S. McAdams, V. Roussarie, A. Chaigne, and B. L. Giordano, "The psychomechanics of simulated sound sources: Material properties of impacted thin plates," *The Journal of the Acoustical Society of America*, vol. 128, no. 3, pp. 1401–1413, 2010.

[11] M. Houel, A. Arun, A. Berg, A. Iop, A. Barahona-Rıos, and S. Pauletto, "Perception of emotions in knocking sounds: An evaluation study," in *17th Sound and Music Computing Conference, Online*, 2020.

[12] P. N. Juslin and P. Laukka, "Communication of emotions in vocal expression and music performance: Different channels, same code?" *Psychological bulletin*, vol. 129, no. 5, p. 770, 2003.

[13] F. Takahashi and Y. Kawabata, "The association between colors and emotions for emotional words and facial expressions," *Color Research & Application*, vol. 43, no. 2, pp. 247–257, 2018.

[14] P. Valdez and A. Mehrabian, "Effects of color on emotions." *Journal of experimental psychology: General*, vol. 123, no. 4, p. 394, 1994.

[15] C. Mohr, D. Jonauskaite, E. S. Dan-Glauser, M. Uusküla, and N. Dael, "Unifying research on colour and emotion: time for a cross-cultural survey on emotion associations to colour terms," *Progress in Colour Studies: Cognition, language and beyond*, pp. 209–222, 2018.

[16] R. B. Hupka, Z. Zaleski, J. Otto, L. Reidl, and N. V. Tarabrina, "The colors of anger, envy, fear, and jealousy: A cross-cultural study," *Journal of cross-cultural psychology*, vol. 28, no. 2, pp. 156–171, 1997.

[17] J. H. Xin, K. Cheng, G. Taylor, T. Sato, and A. Hansuebsai, "Cross-regional comparison of colour emotions part i: Quantitative analysis," *Color: Research & Application*, vol. 29, no. 6, pp. 451–457, 2004.

[18] M. R. Zentner, "Preferences for colours and colour–emotion combinations in early childhood," *Developmental Science*, vol. 4, no. 4, pp. 389–398, 2001.

[19] B. Manav, "Color-emotion associations and color preferences: A case study for residences," *Color: Research & Application*, vol. 32, no. 2, pp. 144–150, 2007.

[20] K. Naz and H. Epps, "Relationship between color and emotion: A study of college students," *College Student Journal*, vol. 38, no. 3, p. 396, 2004.

[21] M. Hemphill, "A note on adults' color–emotion associations," *The Journal of genetic psychology*, vol. 157, no. 3, pp. 275–280, 1996.

[22] E. Joosten, G. Lankveld, and P. Spronck, "Colors and emotions in video games," in *11th International Conference on Intelligent Games and Simulation GAME-ON*, 2010, pp. 61–65.

[23] M. M. Aslam, "Are you selling the right colour? a cross-cultural review of colour as a marketing cue," *Journal of marketing communications*, vol. 12, no. 1, pp. 15–30, 2006.

[24] K. Haller, "Colour in interior design," in *Colour Design*, J. Best, Ed. Elsevier, 2017, pp. 317–348.

[25] G. Crippa, V. Rognoli, and M. Levi, "Materials and emotions: A study on the relations between materials and emotions in industrial products," *8th International Conference on Design and Emotion: Out of Control - Proceedings*, 01 2012.

[26] Y. Ebe and H. Umemuro, "Emotion evoked by texture and application to emotional communication," in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, 2015, pp. 1995–2000.

[27] M. Iosifyan and O. Korolkova, "Emotions associated with different textures during touch," *Consciousness and cognition*, vol. 71, pp. 79–85, 2019.

[28] A. Steinvall, "Colors and emotions in english," in *Anthropology of colour*, R. E. MacLaury, G. V. Paramei, and D. Dedrick, Eds. John Benjamins Publisher, 2007.

[29] R. D'Andrade and M. Egan, "The colors of emotion," *American ethnologist*, vol. 1, no. 1, pp. 49–63, 1974.

HOW DOES THE SPOTIFY API COMPARE TO THE MUSIC EMOTION RECOGNITION STATE-OF-THE-ART?

Renato PANDA (panda@dei.uc.pt)[1, 2], **Hugo REDINHO** (redinho@student.dei.uc.pt)[1], **Carolina GONÇALVES** (mcarolina@dei.uc.pt)[1], **Ricardo MALHEIRO** (rsmal@dei.uc.pt)[1, 3] and **Rui Pedro PAIVA** (ruipedro@dei.uc.pt)[1]

[1]*CISUC*, *DEI*, **University of Coimbra**, Portugal
[2]*Ci2 – Smart Cities Research Center*, **Instituto Politécnico de Tomar**, Tomar, Portugal
[3]**Miguel Torga Higher Institute**, Coimbra, Portugal

ABSTRACT

Features are arguably the key factor to any machine learning problem. Over the decades, myriads of audio features and recently feature-learning approaches have been tested in Music Emotion Recognition (MER) with scarce improvements. Here, we shed some light on the suitability of the audio features provided by the Spotify API, the leading music streaming service, when applied to MER. To this end, 12 Spotify API features were obtained for 704 of our 900-song dataset, annotated in terms of Russell's quadrants. These are compared to emotionally-relevant features obtained previously, using feature ranking and emotion classification experiments. We verified that energy, valence and acousticness features from Spotify are highly relevant to MER. However, the 12-feature set is unable to meet the performance of the features available in the state-of-the-art (58.5% vs. 74.7% F1-measure). Combining Spotify and state-of-the-art sets leads to small improvements with fewer features (top5: +2.3%, top10: +1.1%), while not improving the highest results (100 features). From this we conclude that Spotify provides some higher-level emotionally-relevant features. Such extractors are desirable, since they are closer to human concepts and allow for interpretable rules to be extracted (harder with hundreds of abstract features). Still, additional emotionally-relevant features are needed to improve MER.

1. INTRODUCTION

During the beginning of the 21st century notorious changes have occurred on the way people access and consume music, movies and other media. Before technical advances such as ubiquitous internet access, digital compact audio formats or the massification of mobile devices, music consumption would swirl around physical media such as tapes, optical discs or vinyl. These were normally sold at local stores, which provided limited and normally region-specific catalogs. In the course of the last decade this paradigm has changed, with music access, available through streaming services, taking over music ownership.

The first internet-based music services, centered on illegal distribution, contributed to the decline of music revenues, which achieved its lowest point in 2014. Since then, music revenues have risen for 6 consecutive years[1] led by streaming services such as Spotify, Deezer, or Pandora [1].

Nowadays these services provide easy access to millions of songs with unprecedented convenience (e.g., Spotify offers over 70 million tracks as of December 31, 2020[2]). However, such massive amount of data requires better search and discovery mechanisms than simply searching by artist, title or genre. Spotify mitigates these issues by using several data-driven personalization methods, and manually curated playlists. These are mostly based on users' listen history, while less focus has been given to the audio content due to the complexity of the task.

Meanwhile, we know that music is a language to express emotions, with some considering it to be its primary function [2]. Thus, music emotion recognition (MER) researchers have been proposing computational models to uncover and exploit these relations automatically. It is known that Spotify and other industry players are interested in various music information retrieval (MIR) topics, e.g., Spotify is said to be planning to use the user tone to detect his/her mood and personalize music suggestions[3].

So, how does the MER current state-of-the-art research compare with the solutions of the aforementioned services? In this paper we explore this question and shed some light on possible paths for future research. To this end, we assess how the audio features provided by the Spotify API[4] compare with the features used in MER. First, 12 Spotify audio features were gathered for a subset of 704 songs from our 900-song dataset [3]. Several audio feature ranking and emotion classification experiments are then run using these, as well as the top 100 features identified experimentally by our team as emotionally-relevant in [3], to understand how these compare and complement.

Among others, we verified that the 12 Spotify features are worse at discriminating Russell's quadrants, although four of them proved relevant. While not being a magic

[1] With an expected drop in revenues due to COVID-19 [1]
[2] https://newsroom.spotify.com/company-info/
[3] https://www.bbc.com/news/entertainment-arts-55839655
[4] https://developer.spotify.com/documentation/web-api/

wand, the inclusion of extra audio features by Spotify might help improving their recommendation data, something that may even be already in place internally, while not available thought their public API.

This paper is organized as follows. Section 1 introduces the problem, motivation and objectives. Section 2 briefly describes the related work relevant to the topic. Next, the experimental setup, including dataset, audio features and classification strategies are presented in Section 3, while Section 4 discusses the observed results. Finally, Section 5 draws the conclusions and suggests future work.

2. RELATED WORK

Music Emotion Recognition is a subfield of MIR which aims to automatically extract emotional information present in music. The field bridges knowledge from areas as diverse as music theory, machine learning, digital signal processing and psychology. In very broad terms, a typical MER solution uses a source of musical data (e.g., audio signals, lyrics, scores) to understand the governing relations between its properties (i.e., features) and emotional cues (e.g., annotations, in supervised learning), deriving rules to classify the emotion in newer examples (i.e., unknown musical data) [4].

The relations between music and emotions have been a matter of study for long by psychologists, with many relations documented [5]. As an example, consonant harmonies and major modes are usually associated with positive emotions, while the opposite with negative ones. However, the mechanisms governing these are not fully understood yet, and some studies present contradictory results [5].

In the field of computer science, several digital signal processing algorithms have been developed to capture sound and music characteristics from audio signals. These have been originally proposed to solve specific problems (e.g., speech recognition) but soon were employed in other MIR subfields, with MER being no exception [6].

In this journey of recognizing emotions with computers, different emotion paradigms (e.g., categorical or dimensional) and related taxonomies (e.g., Hevner [7], Russell [8]) have been used. These, intertwined with one of the datasets proposed in the field (e.g., [9]–[11]) served as foundations to many works. From emotion classification using raw audio signals [3], [12], [13], symbolic notations [14], or lyrics [9], to multi-label classification (i.e., several emotions per clip) [15], [16], dimensional MER [10], music emotion variation detection [11], [17] or multi-modal approaches [13], [14].

The majority of these works follow a typical machine learning approach, with handcrafted features, testing different datasets and machine learning strategies. However, results have stagnated over time, with authors suggesting better solutions "should perhaps be more musical knowledge-intensive" [18] to narrow the so-called semantic gap [19], a view also supported by us [3].

An alternative (or even complementary) path to the approach with handcrafted features is deep learning (DL). DL has been gaining momentum due to the ever increasing computational power and big data. There, AI-powered feature engineering is used, by feeding the neural network directly with the dataset (e.g., in the form of spectrograms). Several MER studies have tested techniques such as convolutional and recurrent neural networks [17]. However, such solutions fell short of expectations (so far), in part due to the lack of massive high quality datasets, which are complex to obtain [3] – one of the open problems in MER.

2.1 Spotify

Spotify is the major music streaming service, with 345 million monthly active users and 155 million subscribers in 93 markets as of December 31, 2020[5]. The service is responsible for reshaping the way music listeners experience music nowadays, exchanging ownership for easier access to large catalogs and personalized recommendations. Nowadays the service revolves around playlists, with 4 billion of them interconnecting 70+ million songs.

With such massive amounts of data, the company was able to expand from a simple music streaming player to a data-driven personalization service that drives discovery and engagement, increasing in value with each new user. The effect it now has on new songs, artists and their earnings is immense, e.g., "being added to Today's Top Hits, a list with 18.5 million followers (…), raises streams by almost 20 million and is worth between $116,000 and $163,000" in additional revenue from Spotify alone [20]. Still, the question remains: how exactly does Spotify recommend personalized content to each user?

Although not fully documented to the public, several details are known. Traditionally, recommendations relied on collaborative filtering [21], a technique used to understand a specific user's music taste based on historical listening data from all users, for instance using implicit matrix factorization [22]. Such techniques are content-agnostic (i.e., do not use the audio signals), relying only on users' consumption patterns. This fact is also its Achilles heel: they are unable to recommend new and unpopular songs given the lack of listening data – known as the *cold-start problem*, relevant when ~40,000 tracks are added daily[6].

To overcome this, Spotify includes additional sources of information, driven by the acquisition of The Echo Nest in 2014 – a music intelligence service providing automatic data extraction from songs by web crawling (e.g., metadata, lyrics, reviews), and digital signal processing techniques on the audio signal itself. Among others, it is able to estimate the danceability of a song or its valence.

Moreover, Spotify is known to use deep learning techniques to crunch metadata and audio signals for better recommendations. This began with Dieleman's work with deep convolutional neural networks [23] and has been evolving since then. Nowadays Spotify provides datasets and contributes with research in diverse areas from recommendation to user modeling or music creation[7]. Despite their advances, one major hurdle persists regarding MER

[5] https://newsroom.spotify.com/company-info/

[6] https://haulixdaily.com/2019/05/spotify-40000-tracks-per-day

[7] https://research.atspotify.com/

– the absence of quality emotion annotations that can help to better understand and predict what makes a happy or sad song.

3. EXPERIMENTS

The system provided by Spotify has been evolving over the years, fusing different sources of data (i.e., historical usage data, music metadata, web crawling, high-level features from The Echo Nest and DL). Even so, it is unclear if the emotional content present in the audio signal is being captured. To shed some light on this, we tested the high-level features provided by the Spotify API in a typical MER problem and compared them with the state-of-the-art.

In brief, we built on our previous work [3], where a 900-clip dataset was used to predict Russell's quadrants [8], identifying and proposing novel emotionally-relevant features. Here, we adopt a similar strategy, adding the 12 audio features provided by the Spotify API[8]. Each step in this direction is described in the following subsections.

3.1 Dataset

The original dataset contains 900 audio clips (up to 30 sec) annotated with Russell's quadrants (i.e., Q1 to Q4, representing respectively happiness, tension/aggression, sadness, and calmness). Both samples and metadata were sourced from AllMusic[9] and balanced (quadrants and genres). The AllMusic mood tags were matched against Warriner's norms of valence and arousal for English words and transformed into quadrant annotations, following a manual validation by volunteers. Further details in [3].

Crawling the Spotify service returned 704 of the 900 songs, for which audio features were obtained and used.

3.2 Audio Features

In general terms, a feature describes a characteristic part of something, it may be the composer, genre, its tempo, duration, or even more abstract statistics of the signal itself [24]. In this study we use computational audio features proposed in the literature or provided directly by Spotify.

3.2.1 Literature Features

From the audio clips, a total of 2719 features were initially extracted using MIR Toolbox, Marsyas and PsySound3 audio frameworks, as well as a set of novel features proposed in our previous work [3]. This high number is caused by the duplication of features across frameworks, as well as the summarization of time series into several statistics. These were then reduced by excluding features whose values had zero variance, as well as pairs of features with correlation higher than 0.9, as detailed in [3].

Next, the ReliefF algorithm [25] was used to identify and rank features according to their emotional relevance. Then, emotion classification experiments with the top 100 of these features achieved an F1-measure of 76.4% [3].

In this work we use these 100 features for the subset of 704 songs[10].

3.2.2 Features provided by the Spotify API

Spotify provides 12 audio features through its API[11]:

- Acousticness – whether the track is acoustic
- Danceability – how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity
- Energy – a perceptual measure of intensity and activity based on dynamic range, perceived loudness, timbre, onset rate, and general entropy
- Instrumentalness – whether a track contains no vocals (> 0.5 indicate instrumental tracks, with confidence increasing as it approaches 1.0)
- Key – the key the track is in according to standard Pitch Class notation
- Liveness – indicates presence of an audience in the recording (> 0.8 provides strong likelihood that the track is live)
- Loudness – the overall loudness in decibels (dB)
- Mode – the modality (major or minor) of a track
- Speechiness – presence of spoken words
- Tempo – overall estimated tempo in beats per minute (BPM)
- Time signature – estimated overall time signature (meter) of a track
- Valence – describes the musical positiveness conveyed (higher valence sounds more positive, e.g. happy, cheerful)

These 12 high-level audio features were obtained for our 704 tracks and used in the experiments described next.

3.3 Feature Selection and Emotion Classification

Having the audio features, our next step was to understand which of these were more suited to our quadrants classification problem by using the ReliefF [25] algorithm. Several reasons weighted in favor of this particular algorithm, namely, not being as CPU intensive as forward feature selection (which performs exhaustive classification tests) and the fact that it provides a rank / weight for each feature. At each iteration of the algorithm, a random song is selected and the K closest songs of the same class (i.e., quadrant) and of different classes are picked (using the Euclidean distance between feature vectors). The weight of each feature is then adjusted based on this distance between same vs. different class instances. Three different rankings were computed:

1. Ranking of our 100 features
2. Ranking of the 12 Spotify API audio features
3. Ranking of all 112 features combined

[8] https://developer.spotify.com/documentation/web-api/

[9] https://www.allmusic.com/

[10] https://github.com/renatopanda/TAFFC2018

[11] https://developer.spotify.com/get-audio-features/

The main goal with the first one was to re-rank and understand the changes in our top 100 features, since the original order was computed with the complete dataset (900 songs), against 704 now. The second ranking helps understanding which Spotify API features are more emotionally-relevant and better discriminate our dataset. Finally, ranking the 112 features together gives us a better understanding on how they compare and work together.

Using these rankings, several classification tests were run. Here, Support Vector Machines (SVM) was selected given its better results in the MER field [4] and our prior experience [3]. To this end, John Platt's implementation of sequential minimal optimization (SMO) for training SVMs, provided by the Weka[12] framework, was used with 10 repetitions of 10-fold cross-validation.

4. DISCUSSION

This section discusses the outcomes of the experiments described previously, regarding the dataset reduction, feature ranking and emotion classification[13].

4.1 Dataset Analysis

Our original dataset contains 900 audio clips, balanced across quadrants (225 clips each) and genres. However, 196 of these were not found in Spotify. The missing songs are spread across all quadrants, as shown in Figure 1, with Q2 (tense/anxious) affected the most (58 clips eliminated, now 167) and Q4 (calm/relaxed) on the other end of the spectrum (lost 37, now with 188).

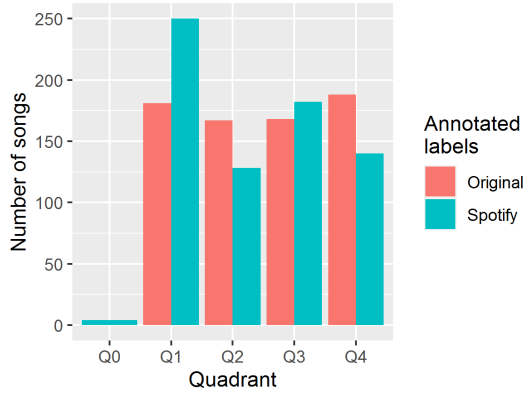


Figure 1. Distribution of the 704 clips across quadrants (Q0 are songs with neutral energy and/or valence).

Spotify estimates both energy and valence (EV) of each song. These are the two dimensions that define Russell's circumplex model of emotion [8]. In reality, energy is not exactly arousal, but serves as its surrogate. By transforming them into quadrants we can understand song distribution across quadrants (Figure 1). As illustrated, Spotify seems skewed towards Q1, with 250 of 704 songs considered happy. On the other hand, only 128 end up in Q2 (tense, anxious).

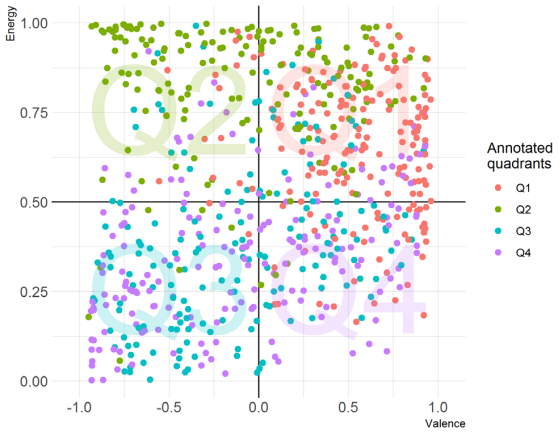


Figure 2. Songs according to Spotify's EV values (colored according to the original dataset annotations).

To better grasp these differences, Figure 2 presents the 704 songs placed in the Russell's plane, colored according to the original annotations. A perfect scenario would be each color (original quadrant) contained inside a single plane quadrant (Spotify quadrant), which is far from observed. Despite the lack of accuracy, this visualization uncovers interesting tendencies. Namely, the energy metric is more accurate, since most songs originally tagged with Q1 and Q2 (red and green) are placed in the top half of the plot, while the remaining are in the opposite end. The same cannot be said about valence, since, for instance, many Q2 songs (anxious/tense, in green) have positive valence, and Q4 songs (calm/relaxed, in purple) negative valence.

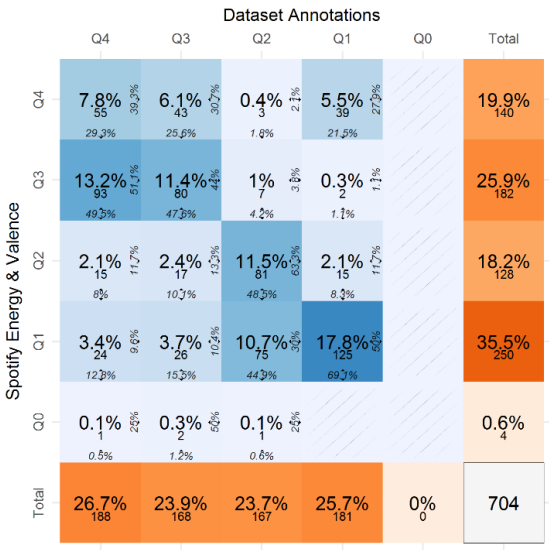


Figure 3. Original quadrants (annotations) vs. Spotify energy-valence based quadrants confusion matrix (each tile showing counts, overall percentage, row percentage and column percentage).

[12] https://www.cs.waikato.ac.nz/ml/weka/

[13] Data available at: https://github.com/renatopanda/SMC2021

These differences were somewhat expected given that: i. Spotify values (i.e., energy and valence) are computed from the audio signal, while the original dataset was validated by humans, and ii. predicting valence from audio is an harder problem, still to be fully addressed in MER [3]. In other words, it demonstrates that existing audio feature extractors still need to be further studied and improved.

To conclude, Figure 3 provides a more analytical view of this. Here we can see that Q1 is where we find more agreement (125 clips), while Q2 songs (originally 167) are mostly spread by Spotify between Q1 (75) and Q2 (81). A similar situation happens with Q3 and Q4, with the majority of these placed in Q3 by Spotify (93 + 80).

4.2 Standard Features' Ranking and Classification

The next step was to understand how the elimination of songs from the dataset (196 songs dropped) influenced both features' relevance and emotion classification.

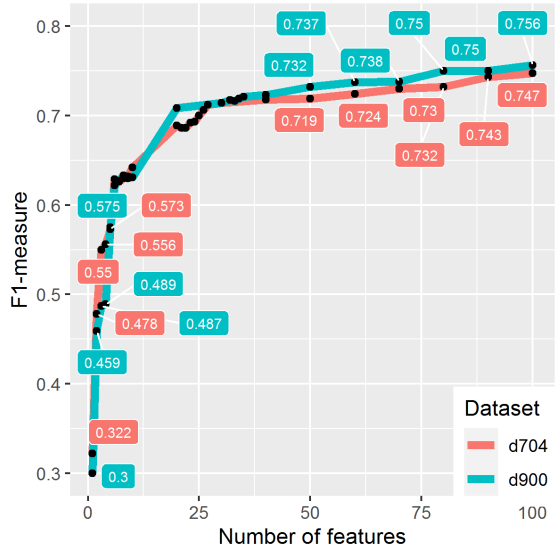


Figure 4. Classification results for 900 vs. 704 instances using the obtained feature ranking.

To this end we used the Weka ReliefF implementation to re-rank the top 100 features obtained previously using the 704 songs subset. Given the nature of ReliefF, several differences in the feature order were expected, and several reasons contribute to this. First, one-fifth of the songs were removed and ReliefF estimates the weight of each feature based on the Euclidean distance between features of randomly picked instances' assigned to distinct classes. Removing instances from the experiment will lead to different distances and thus different weights. Moreover, a different number of instances or even the seed (random) will lead to slight weight variations and ranking fluctuations (e.g., the top 10 features of each ranking might be different). Thus, the new ranking is useful to assess how the classification changes by using the best N features (e.g., top 10 to top 100), more than to assess which feature comes first.

Using this ranking, quadrants classification for both datasets (900 and 704) was tested using SVMs and is shown in Figure 4 ($C = 8$, $\gamma = 0.08$ for 704, $C = 7$, $\gamma = 0.1$ for 900).

As illustrated, the results are very close between the two sets, with the highest F1-measures separated by less than 1% (75.6% to 74.7%). The slightly lower result of 704 subset (with 21.8% fewer songs) might be indicative of the impact that the size of the dataset has in machine learning problems – the more quality data available, the better, more generalizable and robust are the identified patterns.

4.3 Spotify API Features' Ranking and Classification

Although only 12 features are provided, these are of much higher level than most features found in audio frameworks. As an example, Spotify's danceability is derived by combining information about tempo, rhythm, beat strength and regularity. Results of the ReliefF algorithm in our dataset (704) and the 12 features, in Figure 5, sheds some light on their contribution to discriminate across quadrants.

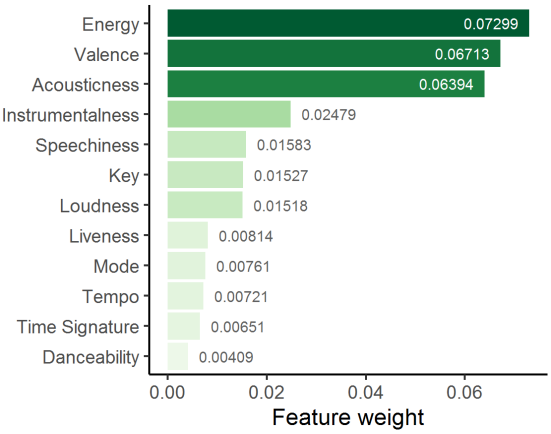


Figure 5. Influence of Spotify API features to separate songs among Russell's quadrants, according to ReliefF.

As expected, energy and valence audio features have the highest weight to the problem. These features are highly correlated with the quadrants, since they define the Russell's plane. If the algorithms that extract them from audio signals were perfect, they would probably be enough. Still, using only these in classification experiments achieves an F1-measure of 47.8%. In addition, the acousticness feature was also highly rated and thus required further inspection, as illustrated in Figure 6. Originally developed by The Echo Nest[14], it distinguishes between natural acoustic sounds (e.g., acoustic guitar, piano, unprocessed human voice – high acousticness) and mostly electric sounds (e.g., electric guitars, synthesizers, auto-tuned vocals – low).

Further analysis of the acousticness feature in our dataset uncovered two interesting facts. First, low acousticness,

[14] https://blog.echonest.com/post/53511313353/new-audio-attribute-acousticness

indicating mostly electric sounds, is more prevalent in Q1 (i.e., happy) and especially Q2 (i.e., tense/angry) songs (Figure 6-A). Secondly and as a result of the first, a high (negative) correlation was found between energy and acousticness (Figure 6-B).

Still regarding feature weights, on the other end of the spectrum we have features such as danceability, time signature, tempo and mode. While a lower weight for time signature (and also key) is not a surprise, one would expect features like mode [6] and danceability to have a different result. After all, major modes are usually associated with happiness and, intuitively, one would expect danceability to be too. In this case, it may happen that the algorithms used are not robust enough yet or caused by specificities of this dataset. In addition, it may be that these features work better in the presence of others currently missing.

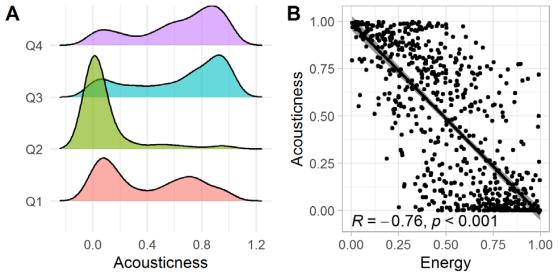


Figure 6. Distribution of acousticness values per quadrant (A), and correlation between energy and acousticness (B).

Figure 7. Classification results using the 12 Spotify API features (our top10 features added for comparison).

The classification results obtained using the 12 Spotify API features is presented in Figure 7, with the results from our best 10 features added for comparison. One notable fact is that with only one feature – energy, the classifier is able to correctly identify more than half of the songs, achieving an F1-measure of 44.2% (and accuracy of 50.9%), while our best feature is way behind (32.2%).

However, given the abovementioned findings (i.e., less relevant features and high correlations), the classification results using Spotify API features scarcely increase from the fourth to the twelfth feature.

4.4 Combining both sets of Features

After testing each set of features individually – Spotify API and ours, the logical step was to verify if the combination of both would improve the classification results. To this end, we combined both into a 112 features set and applied the ReliefF feature selection algorithm to assess the weight of each feature to the problem. Remarkably, four out of the 12 Spotify API features were among the top 5, while another four were placed in the bottom 5, as illustrated in Figure 8. The remaining were ranked in positions 52nd, 74th, 92nd, and 99th.

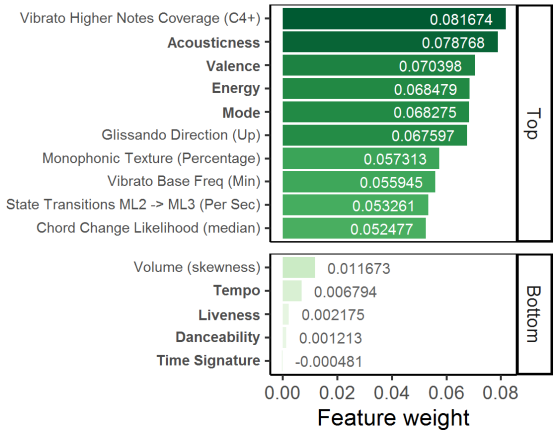


Figure 8. Best and worst rated features according to ReliefF (Spotify API features in **bold**).

In addition, the top 10 features included three features related to expressive techniques, two related to musical texture (proposed by us in [3]) and one to melody (from PsySound3). As expected, the three most important Spotify API features were again energy, valence and acousticness. Though, this time their order was reversed and each had slightly different weights. Moreover, mode was ranked fifth amongst the 112 features, obtaining a much higher weight than previously obtained when only 12 features were ranked (in Section 4.3).

To understand these differences we need to remember that ReliefF is a filter model – uses statistics extracted from the training data (i.e., Euclidean distance) and correlates them with the associated labels (i.e., quadrants). In brief, in each iteration a random instance (song) is selected and the two songs closer to it (one for each class) are also selected. This is based on the Euclidean distance between feature vectors, as if they were placed in an N-dimensional space. Then, the weight of each feature is readjusted using the differences between the feature values of the random song and the other two (of same and different class). This has several implications, two of those are: 1) a different number of features (dimensions) will influence the distance between instances and 2) the random selection of

songs to compute the weights (non-exhaustive) will lead to slight variations in feature weights over each run of the algorithm (with different random seeds). Both points shed some light on the slight order and weight differences, and might explain the increase in the importance of mode. Since mode is binary (major or minor), its relevance was lower when combined with other 11 features – especially considering that only 3 were of high relevance. Once the dimensionality increases and the number of relevant features increases, its own (i.e., mode) discrimination power is increased (interaction between features).

The next step was to use the combined ranking and repeat the classification experiments with SVMs (C = 0.07, γ = 4). As summarized in Table 1, no substantial differences were observed between results of our set of 100 features and the combined set of 112 features. Here, the more noteworthy were: 1) the higher F1-measure from Spotify API when only one feature is used; 2) the fact that with few features (less than 10), the combined set performs slightly better; and 3) the convergence in results as the amount of features increases (illustrated in Figure 9).

No Features	Spotify	Ours	Combined
1	44.2%	32.2%	31.4%
5	56.2%	57.3%	59.6%
10	56.1%	64.2%	65.3%
20	-	68.9%	68.9%
50	-	71.9%	71.8%
Best result (# features)	58.5% (11)	74.7% (100)	74.2% (100)

Table 1. Summary of the emotion classification results.

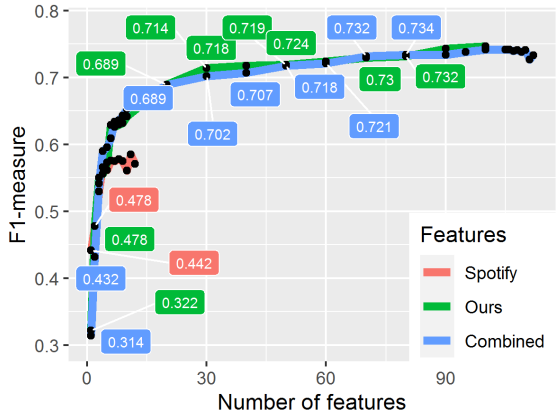


Figure 9. Classification results for each feature set.

The first is a consequence of the selected ranking algorithm, since the same feature is available in both Spotify and Combined sets (energy) but was only selected as first in the former. Although much more resource-intensive, one approach to mitigate this is by combining ReliefF (filter model type of feature selection) with a wrapper model (e.g., forward feature selection). Still, this is dissipated when more features are added. When the number of features in use increases (e.g., 5 to 10), we see the advantages of combining both sets.

As the number of features in use increases, the performance differences become negligible. One possible explanation is that the combination of some of the (lower-level) features in our top 100 may be able to capture similar emotional cues to the few higher-level proposed by Spotify API and thus their relevance drops in the combined set. As an example, the most relevant Spotify API feature (energy) combines dynamic range, perceived loudness, timbre, onset rate, and general entropy. Such characteristics were previously extracted separately to obtain our top 100 feature set [3].

5. CONCLUSIONS

This paper offered an analysis on the suitability of the Spotify API audio features to the music emotion recognition field. As part of this, its features were compared to audio features previously proposed in the scientific literature that are known to be emotionally-relevant.

From the experiments, several conclusions were drawn. First, three of the 12 Spotify API features were identified as highly relevant to emotion classification – energy, valence and acousticness. While the first 2 were expected, given their relation to the Russell's plane, acousticness – which was found to be highly correlated with energy, was somewhat new. Moreover, tense/anxious songs were almost exclusively low on acousticness (i.e., non-natural, electric sounds), information that complements our previous survey on emotionally-relevant audio features [6].

Secondly, the 12 features provided by the Spotify API are subpar to the problem in analysis, achieving only 58.5% F1-measure, when compared to the state-of-the-art (74.7%). While Spotify's goal is music recommendation and user taste modeling, we believe that the addition of emotionally-relevant audio features may improve the system, after all some argue that music's primary function is to express emotions [2]. Such idea might even be already in use, but just not exposed in their public API.

Finally, we believe that novel audio feature extractors, are needed to improve this as well as other MIR problems, since most MIR solutions are generic, "without relying on musically meaningful features" [18]. These novel features should be higher-level (i.e., closer to human knowledge), providing ways to uncover interpretable rules between emotions and an handful of audio cues, after all that is science's main goal – to explain and understand.

Acknowledgments

This work was supported by CISUC (Center for Informatics and Systems of the University of Coimbra). Renato Panda was supported by Ci2 - FCT UIDP/05567/2020.

6. REFERENCES

[1] IFPI, "Global Music Report 2020: The Industry in 2019," 2020. [Online]. Available:

http://www.idf.org/node/26452?language=es.

[2] A. Pannese, M.-A. Rappaz, and D. Grandjean, "Metaphor and music emotion: Ancient views and future directions," *Conscious. Cogn.*, vol. 44, pp. 61–71, Aug. 2016, doi: 10.1016/j.concog.2016.06.015.

[3] R. Panda, R. Malheiro, and R. P. Paiva, "Novel Audio Features for Music Emotion Recognition," *IEEE Trans. Affect. Comput.*, vol. 11, no. 4, pp. 614–626, Oct. 2020, doi: 10.1109/TAFFC.2018.2820691.

[4] X. Yang, Y. Dong, and J. Li, "Review of data features-based music emotion recognition methods," *Multimed. Syst.*, pp. 1–25, 2017, doi: 10.1007/s00530-017-0559-4.

[5] A. Gabrielsson and E. Lindström, "The Role of Structure in the Musical Expression of Emotions," in *Handbook of Music and Emotion: Theory, Research, Applications*, P. N. Juslin and J. A. Sloboda, Eds. Oxford University Press, 2011, pp. 367–400.

[6] R. Panda, R. M. Malheiro, and R. P. Paiva, "Audio Features for Music Emotion Recognition: a Survey," *IEEE Trans. Affect. Comput.*, pp. 1–1, 2020, doi: 10.1109/TAFFC.2020.3032373.

[7] K. Hevner, "Experimental Studies of the Elements of Expression in Music," *Am. J. Psychol.*, vol. 48, no. 2, p. 246, Apr. 1936, doi: 10.2307/1415746.

[8] J. A. Russell, "A circumplex model of affect," *J. Pers. Soc. Psychol.*, vol. 39, no. 6, pp. 1161–1178, 1980, doi: 10.1037/h0077714.

[9] R. Malheiro, R. Panda, P. Gomes, and R. P. Paiva, "Emotionally-Relevant Features for Classification and Regression of Music Lyrics," *IEEE Trans. Affect. Comput. – TAFFC*, vol. 9, no. 2, pp. 240–254, Apr. 2018, doi: 10.1109/TAFFC.2016.2598569.

[10] Y.-H. Yang, Y.-C. Lin, Y.-F. Su, and H. H. Chen, "A Regression Approach to Music Emotion Recognition," *IEEE Trans. Audio, Speech, Lang. Process. – TASLP*, vol. 16, no. 2, pp. 448–457, 2008, doi: 10.1109/TASL.2007.911513.

[11] A. Aljanaki, Y.-H. Yang, and M. Soleymani, "Developing a benchmark for emotional analysis of music," *PLoS One*, vol. 12, no. 3, Mar. 2017, doi: 10.1371/journal.pone.0173392.

[12] Y. Feng, Y. Zhuang, and Y. Pan, "Popular Music Retrieval by Detecting Mood," in *26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval – SIGIR 2003*, 2003, vol. 2, no. 2, pp. 375–376, doi: 10.1145/860435.860508.

[13] C. Laurier, "Automatic Classification of Musical Mood by Content-Based Analysis," Universitat Pompeu Fabra, 2011.

[14] R. Panda, R. Malheiro, B. Rocha, A. P. Oliveira, and R. P. Paiva, "Multi-Modal Music Emotion Recognition: A New Dataset, Methodology and Comparative Analysis," in *10th International Symposium on Computer Music Multidisciplinary Research – CMMR 2013*, 2013, pp. 570–582.

[15] T. Li and M. Ogihara, "Detecting emotion in music," in *4th International Symposium on Music Information Retrieval – ISMIR 2003*, 2003, pp. 239–240.

[16] B. Wu, E. Zhong, A. Horner, and Q. Yang, "Music Emotion Recognition by Multi-label Multi-layer Multi-instance Multi-view Learning," in *22th ACM International Conference on Multimedia – ACM MM 2014*, 2014, pp. 117–126, doi: 10.1145/2647868.2654904.

[17] M. Malik, S. Adavanne, K. Drossos, T. Virtanen, D. Ticha, and R. Jarina, "Stacked Convolutional and Recurrent Neural Networks for Music Emotion Recognition," in *14th Sound & Music Computing Conference – SMC 2017*, 2017, pp. 208–213.

[18] R. Scholz, G. Ramalho, and G. Cabral, "Cross task study on mirex recent results: An index for evolution measurement and some stagnation hypotheses," 2016.

[19] Ò. Celma, P. Herrera, and X. Serra, "Bridging the Music Semantic Gap," in *Workshop on Mastering the Gap: From Information Extraction to Semantic Representation, held in conjunction with the European Semantic Web Conference*, 2006, vol. 187, no. 2, pp. 177–190.

[20] L. Aguiar and J. Waldfogel, "Platforms, Promotion, and Product Discovery: Evidence from Spotify Playlists," Seville, 2018–04, 2018. doi: 10.3386/w24713.

[21] C. Johnson, "Algorithmic Music Recommendations at Spotify," 2014, [Online]. Available: https://www.slideshare.net/MrChrisJohnson/algorithmic-music-recommendations-at-spotify.

[22] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *IEEE International Conference on Data Mining*, 2008, pp. 263–272, doi: 10.1109/ICDM.2008.22.

[23] A. van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Neural Information Processing Systems Conference (NIPS 2013)*, 2013, pp. 2643–2651.

[24] D. Huron, "What is a Musical Feature? Forte's Analysis of Brahms's Opus 51, No. 1, Revisited," *Online J. Soc. Music Theory*, vol. 7, no. 4, 2001.

[25] I. Kononenko, E. Šimec, and M. Robnik-Šikonja, "Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF," *Appl. Intell.*, vol. 7, no. 1, pp. 39–55, 1997, doi: 10.1023/A:1008280620621.

TRACING BACK MUSIC EMOTION PREDICTIONS TO SOUND SOURCES AND INTUITIVE PERCEPTUAL QUALITIES

Shreyan CHOWDHURY (shreyan.chowdhury@jku.at) [1], **Verena PRAHER** (0000-0001-5466-7829) [1], and **Gerhard WIDMER** (0000-0003-3531-1282) [1,2]

[1] *Institute of Computational Perception*, **Johannes Kepler University**, Linz, Austria
[2] *LIT AI Lab*, **Linz Institute of Technology**, Linz, Austria

ABSTRACT

Music emotion recognition is an important task in MIR (Music Information Retrieval) research. Owing to factors like the subjective nature of the task and the variation of emotional cues between musical genres, there are still significant challenges in developing reliable and generalizable models. One important step towards better models would be to understand what a model is actually learning from the data and how the prediction for a particular input is made. In previous work, we have shown how to derive explanations of model predictions in terms of spectrogram image segments that connect to the high-level emotion prediction via a layer of easily interpretable perceptual features. However, that scheme lacks intuitive musical comprehensibility at the spectrogram level. In the present work, we bridge this gap by merging audioLIME – a source-separation based explainer – with mid-level perceptual features, thus forming an intuitive connection chain between the input audio and the output emotion predictions. We demonstrate the usefulness of this method by applying it to debug a biased emotion prediction model.

1. INTRODUCTION

The quest for interpreting the inner workings of "black-box" models and explaining their predictions has led to many recent advances in the area of explainable AI and is becoming an increasingly important staple of all AI subfields. Not only do model explanations help in enhancing trust in the model in applications where its predictions are critical decisions like creditworthiness or medical diagnosis, but they can also often reveal telling signs of algorithmic bias [1, 2]. While algorithmic decisions in the field of MIR are not as life-critical as medical diagnosis, in today's era of music streaming and recommendations, they can have far-reaching effects on diverse audiences, creators, and artists alike. For instance, machine-learning-based music recommender systems have been shown to exhibit severe biases towards or against certain user groups [3]. Interpretable explanations of these models or their predictions would be extremely helpful for identifying such biases.

Previous work on interpretability in MIR has dealt with tasks such as music tagging using self-attention [4] and transcription using invertible neural networks [5], and post-hoc explanations for music content analysis have been used to understand what a genre classifier [6] or a singing voice detector [7–11] have learnt. More recently, audioLIME has been proposed [12, 13] and has shown promise in explaining tagging models [14] as well as recommendation models [15].

However, explanations of *music emotion recognition* systems have received relatively less attention notwithstanding the importance of this task in the areas of musical analysis and recommendation. [16] used models trained with different combinations of sound sources to deduce the importance of each source to emotion predictions. We previously proposed using an intermediate layer of *mid-level perceptual features* [17] to explain music emotion recognition models through a linear connection between the intermediate and final layers [18]. We followed this up with a two-step explanation approach [19] to further explain the predictions in the mid-level layer using components from the input spectrogram. This two-level scheme used LIME (*Local Interpretable Model-agnostic Explanations*) [13] to construct explanations for the mid-level predictions in terms of specific patches of the input spectrogram (which were obtained via image segmentation). While this gave us the regions in the spectrogram contributing most to a particular prediction, these regions did not hold any musical meaning by themselves. As a result, it is difficult to comprehend these explanations in terms of meaningful or intuitive concepts.

In this work, we bridge this gap by merging the audioLIME method, which uses sound sources as explanatory features, with the approach of mid-level features, to obtain comprehensible explanations from the input audio as well as from the perceptual layer. It thus forms an intuitive connection of hierarchical explanations from low-level constituent sources of audio to the high-level emotion predictions through the intermediary mid-level layer, all of which have a musical interpretation.

We believe that explainability is particularly important for developing better music emotion recognition algorithms since it is often difficult to identify misclassifications and biases in this task because of its inherent subjectivity and inter-annotator variability. As an example of real-life application of our method in understanding the po-

tential cause of bias in an emotion model, we demonstrate how a model that has seen few examples of a genre during training results in a pattern of errors on the test set that contains examples of this genre. We trace this pattern of errors back to a particular mid-level feature and this feature to a particular source in the input. Doing so allows us to predict how the model would change when retrained with a balanced training set. We can then qualitatively verify that the retrained model has in fact changed in the way that we expected from our explanations.

2. TWO-LEVEL EXPLANATIONS

Our proposed two-level system will explain emotion predictions by first tracing them back to the most relevant mid-level features and in a second step explain the intermediate mid-level layer via audio sources. We will first describe each of the parts separately from the lowest level (audio sources) to the highest level (emotion predictions) and then put together all the parts in Section 2.3.

2.1 Explaining via Audio Sources: audioLIME

In order to explain individual mid-level predictions we make use of audioLIME, a recently introduced approach based on LIME [13] for interpreting models in MIR [12, 14]. LIME uses simplified inputs based on a set of human interpretable features (depending on the domain of the task – e.g., superpixels for images) to train a simpler explanation model g in order to explain a more complex, potentially deep model f. Previous approaches based on LIME have used time-, frequency-, or time-frequency segments [7], or segments computed by an image segmentation algorithm [19]. audioLIME introduced a new type of interpretable features: sound sources estimated by a music source separation algorithm. In other words, the audioLIME explanation for a given prediction will be in the form of particular sound sources (and possibly specific temporal segments – which we do not use right now), telling us that it is some sonic aspects of these sources that seem to be influential. In our case, the source separator is the pretrained music source separator spleeter [20], thus the explanatory sound sources will be (what the source separator believes are) individual instruments.

2.2 Explaining via Mid-level Perceptual Features

Mid-level features are perceptual qualities or descriptors that emerge from low-level musical building blocks such as timbre, beat structure, harmony, etc. They are more subjective than the low-level features, thus difficult to model using hand-crafted feature extractors, but musically discernible enough to have many people reach a high agreement in annotations. Examples include qualities such as perceived melodiousness or rhythmic complexity [17]. They, therefore, form a suitable choice for an intermediary to higher-level concepts like emotion.

This idea was first used in [18] to explain emotion predictions. We use a similar approach here, but we use the more recently introduced receptive-field regularized ResNets [21, 22] to model the emotions and the mid-level features. In addition, we learn the emotions and mid-level features from two separate datasets using multi-task learning, i.e., jointly learn to predict mid-level features and high-level emotions, allowing us to be flexible with our train and test domains.

The mid-level layer is the penultimate layer of the model, with a linear transformation between it and the final (emotion) layer, thus making the connection interpretable. It is learnt end-to-end from audio spectrograms by optimizing on the combined loss from the emotion and mid-level layers.

Emotion predictions can be interpreted by looking at *effects plots*, which are visual representations of the contribution of each mid-level feature to the final emotion prediction, calculated as the product of the feature value and the weight joining it to the emotion. These indicate the relative influence of the individual features on the final prediction.

2.3 Putting it All Together: Intuitive Two-level Explanations

For a given prediction, we then work backwards. First, we obtain the mid-level explanation of an emotion by computing the effects. The larger the effect of a mid-level feature, the larger is the contribution of that feature to the emotion prediction. Next, we compute the audioLIME explanations for the mid-level feature with the largest effect (we can in principle compute audioLIME explanations for all features to obtain a more diverse explanation, depending on the application). Given these two explanations, we can describe a prediction as being arrived at by the model due to the explanatory mid-level feature, which is in turn most influenced by the input component given by audioLIME.

3. EXPERIMENTAL SETUP

For our analysis with emotion explanations, we first need to train the "explainable" models, which have the penultimate mid-level layer connecting linearly to the final emotion outputs. This section describes the datasets and the training procedure for such models.

3.1 Datasets

For our experiments we are using three different datasets:

3.1.1 Mid-level Perceptual Features Dataset

The *Mid-level Perceptual Features Dataset* introduced in [17] consists of 5000 song snippets with annotations between 1 and 10 for the mid-level descriptors *melodiousness, articulation, rhythmic complexity, rhythmic stability, dissonance, tonal stability*, and *modality* (called "minorness" here). We use this dataset to train the intermediate layer of our emotion model.

3.1.2 DEAM: Database for Emotional Analysis in Music

The *DEAM* dataset [23] is a dataset of dynamic and static valence and arousal annotations. It contains 1,802 songs (58 full-length songs and 1,744 excerpts of 45 seconds) from a variety of Western popular music genres (rock, pop,

	Arousal		Valence	
	RMSE	R2	RMSE	R2
P(P - bl)	0.23	0.61	0.25	0.41
P(P)	0.25 ± 0.03	0.60 ± 0.10	0.31 ± 0.04	0.40 ± 0.14
D(P)	0.27 ± 0.01	0.50 ± 0.03	0.33 ± 0.00	0.30 ± 0.02
D(D)	0.26 ± 0.01	0.49 ± 0.02	0.22 ± 0.01	0.51 ± 0.04
D+P(P)	0.23 ± 0.01	0.65 ± 0.02	0.28 ± 0.00	0.50 ± 0.02
D+P(D)	0.26 ± 0.01	0.50 ± 0.03	0.23 ± 0.01	0.48 ± 0.02

Table 1: Emotion prediction performance with our "explainable" model trained and tested on difference datasets – P: PMEmo, D: DEAM. The dataset inside the parentheses is the test dataset. The top row is the baseline performance from [16].

electronic, country, jazz, etc). In our experiments, we use the static emotion annotations, which are continuous values between 0 and 10.

3.1.3 PMEmo: Popular Music with Emotional Annotation

The *PMEmo* dataset [24] consists of 794 chorus clips from three different well-known music charts. The songs were annotated by 457 annotators with valence and arousal annotations separately for dynamic and static. In our experiments, we use static labels, which are continuous values between 0 and 1.

3.2 Model Training

The mid-level and emotion model is trained end-to-end using audio spectrograms as inputs and optimizing on the combined loss from the mid-level and emotion layers. The batch size is 16 and contains 8 samples from the Mid-level dataset and 8 samples from either the DEAM or the PMEmo dataset. The loss function is the mean squared error. The learning rate is 10^{-3} with cosine annealing, and we perform early stopping on a validation set as regularization. We use the Adam optimizer [25].

The inputs are log-filtered spectrograms (149 bands) of 40-second audio clips peak normalized and sampled at 22.05 kHz with a window size of 2048 samples and a hop length of 704 samples, resulting in 149×1252-sized tensors. If a clip is longer than 40 seconds, we take a random snippet, and if it is shorter, it is looped to 40 seconds.

The labels are scaled to the range $[-1, 1]$ for all three datasets. Therefore, an RMSE of 0.26 would represent 13% error. We split the train and test sets such that they have mutually exclusive sets of artists. A summary of the emotion prediction performance can be found in Table 1.

4. EVALUATION OF EXPLANATIONS

Essentially, there are three targets for empirical evaluation: the two individual components of our two-level explanation framework, and the final composite explanations produced by the model. Regarding the former, the higher level – explanations of emotion predictions in terms of mid-level perceptual features – has already been discussed at length in our previous paper [18]. We showed how effects plots can give insight into the relative importance of various mid-level qualities. The lower level – using audioLIME to explain mid-level feature predictions via audio sources – is a new concept, and the experiments in the following section are intended to validate it. Empirical evidence for the usefulness of the complete, two-level explanation model, finally, will be presented in the form of a study, in Section 5 below, where we demonstrate how these explanations can help us debug a biased prediction model by gaining insight into what the sources of its problems are.

4.1 Explaining Mid-level Features via Sound Sources

Evaluating the quality of explanations is a hard task since there is no consensus on what makes a good explanation, with a variety of desired aims and properties proposed in literature [26–28]). We build our evaluation of audioLIME explanations for the mid-level layer on two metrics, (a) *fidelity* as proposed by Ribeiro et al. along with LIME [13] and, (b) *complexity*, a recently proposed metric for feature-based model explanations [26].

Fidelity measures how well the local model g (the explainer) approximates the global model f (the model up to the mid-level layer in our case) [13] and is computed using the coefficient of determination between the local and global model's predictions as in the original LIME implementation [1] .

In addition to high fidelity, low complexity is desired. The most complex explanation would be the one where all d features get the same attribution (i.e., all weights $g(f, x)_i$ of the linear explanation model g are the same). The simplest explanation concentrates all attribution on one feature. To measure complexity, a probability distribution P_g is defined:

$$P_g(i) = \frac{|g(f, x)_i|}{\sum_{j \in [d]} g(f, x)_j} \tag{1}$$

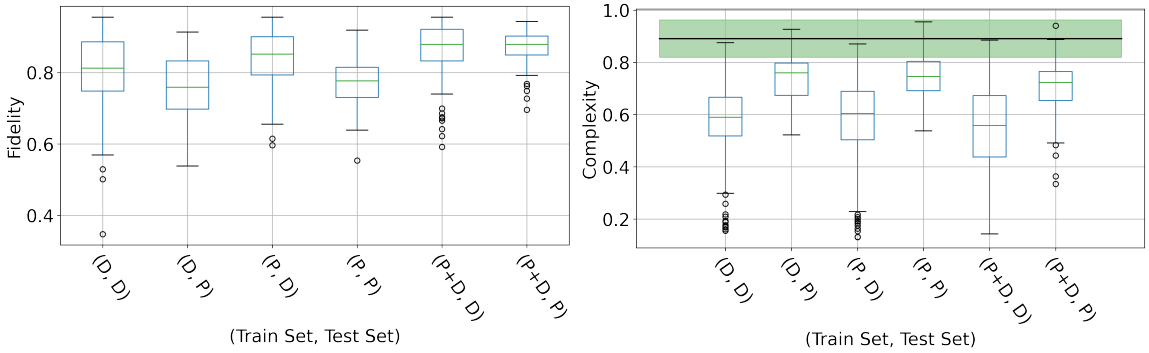
Complexity is then defined as the entropy of this distribution [26]. We compare the complexity per dataset with a random baseline, which is obtained by creating "random" explanations with feature weights drawn from a uniform distribution.

For the analysis, we compute predictions and explanations for all test examples and calculate the above mentioned metrics. The results for one mid-level feature (we picked "rhythmic stability" as it is used later on as an example) are summarized in Figures 1 and 2. We can see in Figure 1a that the fidelity score (coefficient of determination) is relatively high across all combinations of models and test sets. The median score is 0.86 across all explanations (including all mid-level features), the 25%-quantile is at 0.78. This means that for 50% and 75% of the explanations more than 86%, and 78%, respectively, of the variation in the dependent variable (mid-level prediction) can be predicted using the independent variables (instrument sources).

Figure 1b shows the computed complexities, compared to a random baseline. Most explanations are far less complex than the random baseline.

The results shown in the previous figures suggested a relationship between the fidelity and complexity scores.

[1] `https://github.com/marcotcr/lime/`



(a) Fidelity (higher is better).

(b) Complexity (lower is better).

Figure 1: Figure 1a shows the computed fidelity (coefficient of determination R^2 between the predictions by the global model f and the local model g) scores for the evaluated explanations. Figure 1b shows the complexity (entropy of a distribution over the feature attribution weights) scores for the evaluated explanations. The green region shows the standard deviation of complexities for 1000 random explanations, with the black line being the mean.

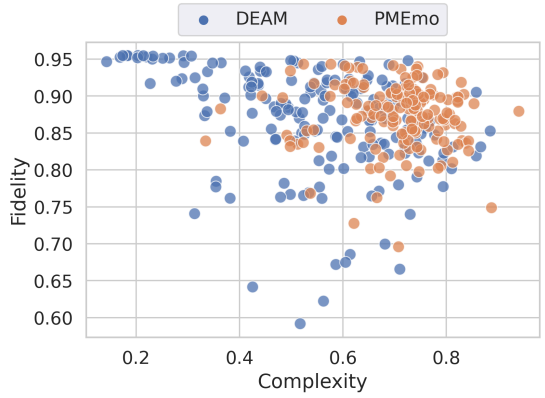


Figure 2: A more detailed view on the relationship between the fidelity score and complexity for the predictions of "rhythmic stability" for a model trained on both datasets. The color indicates the test set.

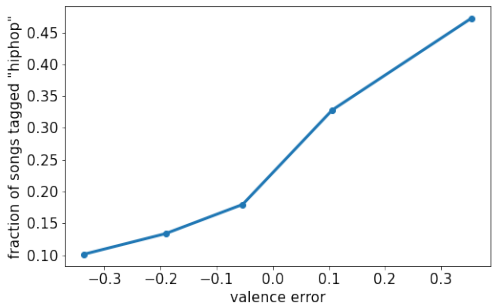


Figure 3: Fraction of hiphop songs in quantiles vs the mean valence error of each quantile over PMEmo dataset (with model trained on DEAM)

Therefore we visualized the two metrics for all explanations computed for "rhythmic stability" for a model trained on the combined data sets in Figure 2. Although they seem related on a dataset level, the metrics do not look related when analyzed for each explanation separately, suggesting that indeed both are needed.

5. MODEL DEBUGGING

A practical use case of our explanation scheme is demonstrated in this section. We use the two-level explanations to understand why an improperly trained model might be overestimating the valence predictions for one particular genre.

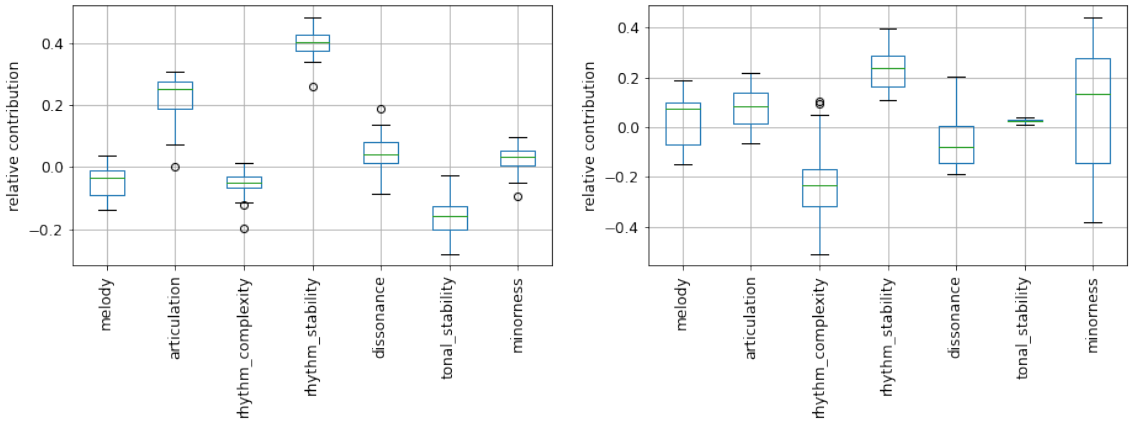
5.1 Setup

First, we use a pre-trained tagger [29] to predict genre tags for all the tracks in the three datasets mentioned in Section 3.1, since we do not have genre metadata for these datasets. This is only done in order to obtain an estimate of the genre-dependence of the emotion predictions later on. We then train two explainable models – one on the DEAM dataset, and one on the combined DEAM and PMEmo dataset. The test set is a fixed but randomly chosen subset of the PMEmo dataset (with a mutually exclusive set of artists from the training set).

5.2 Overestimated Valence for Hiphop

When we take the model trained only on DEAM and use it to predict arousal and valence for the entire PMEmo dataset, we observe that the error in valence shows a pattern – overestimations of valence primarily occur in hiphop songs, as shown in Figure 3.

We can reason about relatively poor performance for hiphop songs based on the discrepancy between the training and testing sets in terms of genre composition. In Figure 5, we can see that PMEmo has a large percentage of hiphop songs whereas both DEAM and Mid-level datasets have a small percentage. Since our model has not seen enough hiphop songs during training, it is to be expected that it does not perform well when it encounters hiphop



(a) Trained on DEAM

(b) Trained on PMEmo+DEAM

Figure 4: Relative effects of the mid-level features for valence prediction for two models trained on different datasets, but tested on the same fixed subset of the PMEmo dataset.

during test. However, a question that is pertinent next is – what is it about hiphop songs that makes our model overestimate their valence?

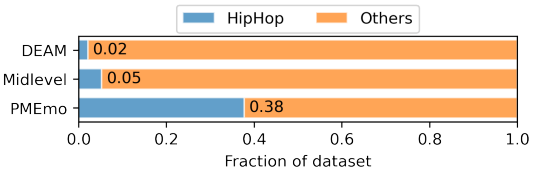


Figure 5: Compositions of datasets as fraction of songs tagged "hiphop" by a pre-trained auto-tagging model [29]

5.3 Explaining Valence Overestimations Using Mid-level Features

To answer this question, we first seek to understand which of the mid-level qualities can be attributed most to high valence predictions. This is the first level of our explanation system. We find these attributions by computing the effects of each mid-level feature on the valence predictions. The effect of a feature is simply the value of that feature multiplied by the weight of the linear connection between it and the target node. In our case, the target is valence and there are seven mid-level features that affect it. We are only interested in relative contribution of each feature, and so we divide each effect by the sum of the absolute values of the effects of all features and take the average across all test songs tagged "hiphop".

We observe that rhythmic stability has the maximum positive relative effect on the prediction of valence. Therefore, we select rhythmic stability for the next step of explanation.

5.4 Explaining Rhythmic Stability Using Sources

Once we have selected a mid-level feature as having the most positive relative effect on the valence, we would like to understand what musical constituents in the input can be attributed to positive contribution to that feature. To do this, we take the help of audioLIME and generate source based explanations for rhythmic stability. The sources available in the current implementation of audioLIME [2] are vocals, drums, bass, piano, and other.

We find that vocals are a major contributing source for the rhythmic stability predictions for the hiphop songs. For songs tagged as other genres, contributing sources are more distributed.

5.5 Re-training the Model with Target Data

Bringing together our two types of explanations, we can reason that the high valence predictions for hiphop songs is due to overestimation of rhythmic stability, which, in this case, can be attributed to the vocals. While there is a lot of diversity in the style of *rapping* (the form of vocal delivery predominant in hiphop), it has been noted that rappers typically use stressed syllables and vocal onsets to match the vocals with the underlying rhythmic pulse [30, 31]. These rhythmic characteristics of vocal delivery (that constitutes "flow", and may add metrical layers on top of the beat) contribute strongly to the rhythmic feel of a song. The positive or negative emotion of hiphop songs is mostly contained in the lyrics – the style of vocal performance does not necessarily express or correlate with this aspect of emotion. Therefore, it makes sense that a model which has seen few examples of hiphop during training should wrongly associate the prominent rhythmic vocals of hiphop to high rhythmic stability and in turn high valence. A model that has been trained with hiphop songs included, we expect, would place less importance on rhythmic stability for the prediction of valence, even if the vocals might still contribute significantly to rhythmic stability. Thus, we expect the relative effect of rhythmic stability for valence to decrease in such a model.

[2] https://github.com/CPJKU/audioLIME

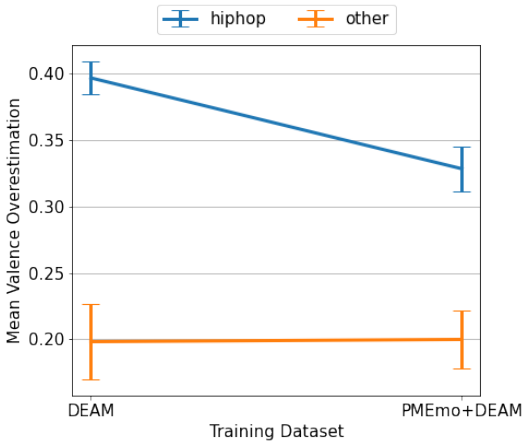


Figure 6: Mean valence overestimations for two models trained on different datasets, but tested on the same fixed subset of the PMEmo dataset.

This is exactly what we observe on a model trained with the combined PMEmo+DEAM dataset. The average relative effects are shown in Figure 4b and we can see that the relative effect of rhythmic stability has decreased while those of minorness, melody, and tonal stability have increased. Thus, the model changed in a way that was in line with what we expected from the analysis of our two-level explanation method.

Looking at mean overestimations (Figure 6) in valence for hiphop and other genres for models trained on DEAM and PMEmo+DEAM shows that valence overestimations of hiphop songs have decreased substantially, without negatively affecting the predictions on other genres [3] .

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed a method to explain music emotion models in an intuitive way using components from low- and mid- levels of the hierarchy of musical concepts by combining audioLIME, which uses input audio sources as explanatory components, with intermediate layer based explanations. We also demonstrated its potential as a tool for model debugging and explaining model behaviour.

This points us towards exploring this method further and getting more granular explanations as a way of improving the effectiveness of this system for MIR. An immediate next step that we are currently pursuing is to extend audioLIME to provide explanations in the form of temporal segments using semantic music segmentation, along with the sound sources.

We are also looking at explaining emotion conveyed in classical piano performances, which pose particular challenges – including the non-availability of training data, where transfer learning of explanatory features becomes necessary [32].

[3] Code for reproducing model debugging experiments is available at https://github.com/shreyanc/model_debugging

Acknowledgments

This work is supported by the European Research Council (ERC) under the EU's Horizon 2020 research & innovation programme under grant agreement No. 670035 ("Con Espressione"), and the Federal State of Upper Austria (LIT AI Lab).

7. REFERENCES

[1] L. Rieger, C. Singh, W. J. Murdoch, and B. Yu, "Interpretations are Useful: Penalizing Explanations to Align Neural Networks with Prior Knowledge," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, vol. 119. PMLR, 2020, pp. 8116–8126.

[2] W. Samek and K. Müller, "Towards Explainable Artificial Intelligence," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, ser. Lecture Notes in Computer Science. Springer, 2019, vol. 11700, pp. 5–22.

[3] M. D. Ekstrand, M. Tian, I. M. Azpiazu, J. D. Ekstrand, O. Anuyah, D. McNeill, and M. S. Pera, "All The Cool Kids, How Do They Fit In?: Popularity and Demographic Biases in Recommender Evaluation and Effectiveness," in *Conference on Fairness, Accountability and Transparency, FAT 2018, 23-24 February 2018, New York, NY, USA*, vol. 81. PMLR, 2018, pp. 172–186.

[4] M. Won, S. Chun, and X. Serra, "Toward Interpretable Music Tagging with Self-Attention," *CoRR*, vol. abs/1906.04972, 2019.

[5] R. Kelz and G. Widmer, "Towards Interpretable Polyphonic Transcription with Invertible Neural Networks," in *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4-8, 2019*, 2019, pp. 376–383.

[6] K. Choi, G. Fazekas, and M. B. Sandler, "Explaining Deep Convolutional Neural Networks on Music Classification," *CoRR*, vol. abs/1607.02444, 2016.

[7] S. Mishra, B. L. Sturm, and S. Dixon, "Local Interpretable Model-Agnostic Explanations for Music Content Analysis," in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, 2017, pp. 537–543.

[8] S. Mishra, E. Benetos, B. L. Sturm, and S. Dixon, "Reliable Local Explanations for Machine Listening," in *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*. IEEE, 2020, pp. 1–8.

[9] S. Mishra, B. L. Sturm, and S. Dixon, "Understanding a Deep Machine Listening Model Through Feature Inversion," in *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR*

2018, Paris, France, September 23-27, 2018, 2018, pp. 755–762.

[10] S. Mishra, D. Stoller, E. Benetos, B. L. Sturm, and S. Dixon, "GAN-based Generation and Automatic Selection of Explanations for Neural Networks," *CoRR*, vol. abs/1904.09533, 2019.

[11] S. Mishra, B. L. Sturm, and S. Dixon, ""What are You Listening to?" Explaining Predictions of Deep Machine Listening Systems," in *26th European Signal Processing Conference, EUSIPCO 2018, Roma, Italy, September 3-7, 2018*. IEEE, 2018, pp. 2260–2264.

[12] V. Haunschmid, E. Manilow, and G. Widmer, "audioLIME: Listenable Explanations Using Source Separation," in *13th International Workshop on Machine Learning and Music*, 2020, pp. 20–24.

[13] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should I Trust You?": Explaining the Predictions of Any Classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. ACM, 2016, pp. 1135–1144.

[14] V. Haunschmid, E. Manilow, and G. Widmer, "Towards Musically Meaningful Explanations Using Source Separation," *CoRR*, vol. abs/2009.02051, 2020.

[15] A. B. Melchiorre, V. Haunschmid, M. Schedl, and G. Widmer, "LEMONS : Listenable Explanations for Music recOmmeNder Systems," in *Advances in Information Retrieval - 43nd European Conference on IR Research, ECIR 2021 (forthcoming), Lucca, Italy, March 28 – April 1*, 2021.

[16] J. de Berardinis, A. Cangelosi, and E. Coutinho, "The Multiple Voices of Musical Emotions: Source Separation for Improving Music Emotion Recognition Models and Their Interpretability," *Proceedings of the 21st International Society for Music Information Retrieval Conference*, no. iii, pp. 310–217, 2020.

[17] A. Aljanaki and M. Soleymani, "A Data-driven Approach to Mid-level Perceptual Musical Feature Modeling," in *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France*, 2018, pp. 615–621.

[18] S. Chowdhury, A. Vall, V. Haunschmid, and G. Widmer, "Towards Explainable Music Emotion Recognition: The Route via Mid-level Features," in *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019*, 2019.

[19] V. Haunschmid, S. Chowdhury, and G. Widmer, "Two-level Explanations in Music Emotion Recognition," *Machine Learning for Music Discovery Workshop, ML4MD at ICML2019*, 2019.

[20] R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: a Fast and Efficient Music Source Separation Tool with Pre-trained Models," *Journal of Open Source Software*, vol. 5, no. 50, p. 2154, 2020, deezer Research.

[21] K. Koutini, S. Chowdhury, V. Haunschmid, H. Eghbal-Zadeh, and G. Widmer, "Emotion and Theme Recognition in Music with Frequency-Aware RF-Regularized CNNs," in *MediaEval 2019 Workshop, Sophia Antipolis, France, 27-30 October 2019*, vol. 2670.

[22] K. Koutini, H. Eghbal-Zadeh, V. Haunschmid, P. Primus, S. Chowdhury, and G. Widmer, "Receptive-Field Regularized CNNs for Music Classification and Tagging," *CoRR*, vol. abs/2007.13503, 2020.

[23] A. Aljanaki, Y.-H. Yang, and M. Soleymani, "Developing a Benchmark for Emotional Analysis of Music," *PloS one*, vol. 12, no. 3, p. e0173392, 2017.

[24] K. Zhang, H. Zhang, S. Li, C. Yang, and L. Sun, "The PMEmo Dataset for Music Emotion Recognition," in *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, ser. ICMR '18. New York, NY, USA: ACM, 2018, pp. 135–142.

[25] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[26] U. Bhatt, A. Weller, and J. M. F. Moura, "Evaluating and Aggregating Feature-based Model Explanations," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, C. Bessiere, Ed. ijcai.org, 2020, pp. 3016–3022.

[27] Z. C. Lipton, "The Mythos of Model Interpretability," *ACM Queue*, vol. 16, no. 3, p. 30, 2018.

[28] N. Tintarev and J. Masthoff, "Evaluating the Effectiveness of Explanations for Recommender Systems - Methodological Issues and Empirical Studies on the Impact of Personalization," *User Model. User Adapt. Interact.*, vol. 22, no. 4-5, pp. 399–439, 2012.

[29] J. Pons, O. Nieto, M. Prockup, E. M. Schmidt, A. F. Ehmann, and X. Serra, "End-to-end Learning for Music Audio Tagging at Scale," in *19th International Society for Music Information Retrieval Conference (ISMIR2018)*, Paris, 2018.

[30] M. Ohriner, "Lyric, rhythm, and non-alignment in the second verse of Kendrick Lamar's "Momma"," *Music Theory Online*, vol. 25, no. 1, 2019.

[31] K. Adams, "On the metrical techniques of flow in rap music," *Music Theory Online*, vol. 15, no. 5, 2009.

[32] S. Chowdhury and G. Widmer, "Towards explaining expressive qualities in piano recordings: Transfer of explanatory features via acoustic domain adaptation," *arXiv preprint arXiv:2102.13479*, 2021.

SLEEP THROUGH SURVEYED: USAGE AND EFFICACY OF STREAMED SOUNDSCAPES CREATED TO HELP INFANTS SLEEP

Lucy WENG (lucy.weng@student.unsw.edu.au)[1], **Adam HULBERT** (a.hulbert@unsw.edu.au)[1], **Emma GIBBS** (Gibbs.Emma@abc.net.au)[2], and **Emery SCHUBERT** (e.schubert@unsw.edu.au)[1]

[1]**UNSW**, Sydney, NSW 2052, Australia
[2]**Australian Broadcasting Corporation**, Sydney, NSW 2007, Australia

ABSTRACT

This paper reports results of a survey that was conducted to assess the use and efficacy of soundscapes composed for an DAB+ radio station and on demand audio App 'ABC-Kids listen' provided by the Australian Broadcasting Corporation (ABC). The soundscapes were a series of previously composed pieces titled *Sleep Through*. 21 people who had listened to one or more of the compositions completed the survey as part of a qualitative study of how music can aid sleep. Results suggested very high overall efficacy, but also revealed applications to situations that did not involve aiding the parents and/or their infants to sleep. These included using *Sleep Through* for pleasure, and for breast feeding. Open-ended responses to the survey were organized into themes labelled: Relaxation (the most prevalent theme), Associations (often linked to the title and environments portrayed in the soundscape), Distraction (strongly related to Relaxation), and Auditory Masking. The theme labelled Habit (using *Sleep Through* to develop healthy sleeping habits) was also considered, but exhibited too great an overlap with other themes. The themes were strongly connected with those found in the literature investigating the use of music for sleep by adults, and also supported the approach of the composer. The study was conducted in collaboration with the composer, with his intentions and responses to the study forming an integral part of the research.

1. INTRODUCTION

The creative process of composition can be a one-directional process, where a composer is charged with both creation and evaluation of the creative work [1]. This responsibility means that feedback from audience, critics and other assessors can be seen as a final judgement on a piece. However, another approach is to collaborate with empirical researchers experienced in designing studies concerned with aesthetic perception. For example, the American composer Roger Reynolds collaborated with several researchers investigating the perception of his composition *The Angel of Death*, producing among other things a special issue in the journal *Music Perception* [2, 3]. Several innovative developments in such projects see productive collaboration between creative and empirical researchers. It is in this rather novel tradition that we report a study on a series of compositions by an Australian composer (author AH) that was released to a global audience via Australia's national broadcaster and provider of online services, the Australian Broadcasting Corporation (ABC).

This paper reports the background to the piece, and the details and results of a survey completed by users of the ABC Kids listen App. The present investigation particularly focuses on an important issue in health and wellbeing, the matter of getting a 'good night of sleep', quality sleep having important health ramifications for both adults [4] and infants [5]. Furthermore, the ABC had a target audience for this project of an important but rarely considered demographic – young children and their carers –and the challenges of getting such a family unit to *Sleep Through* the night, using specially created soundscapes. The use of soundscapes to aid sleep is not new [6], but the close collaboration between music psychology and composer promises new insights. The paper commences with the composer's perspective of the compositions, followed by a survey gathering usage and efficacy information from an online audience. The paper then concludes by reconciling the findings of the survey with the composer's perspectives.

2. COMPOSER PERSPECTIVE

From its inception, the ABC Kids Listen service adopted a child-centered approach, focusing content directly to the young listener without commercial interruption. When commissioned to provide music to support this audience, the consideration was one of how to use broadcast audio to co-create, with the intended listeners, the experience of an acoustic space that was conducive to and supportive of listening, both prior to and during sleep.

In the span of 3 years, approaches evolved over the three iterations of the projects; however, four foundational approaches remained important throughout: (a) Auditory Masking, (b) Relaxation through Entrainment (c) Acoustic space and Incorporation, and (d) Association and Familiarity.

Auditory masking was important to exclude sudden unwanted intrusions that may impact sleep [7]. White noise

is particularly efficacious for this, and although used in maternity wards for this reason, it was deemed unsuitable for broadcast to a wider public and potentially disruptive for co-listening with adult carers. Instead, field recordings using naturally occurring noise (such as waterfalls, streams and wind) were adopted, to align with hypothesised listener expectations for relaxing acoustic environments. These were often accompanied with stacked harmonics produced by synthesis and reflectionless reverberation to further fill the frequency spectrum.

To support a drift into sleep, each of the pieces followed a philosophy of repetition with variation, with the intention of producing a sound event that evokes the experience of 'everyday'—as opposed to 'musical'—listening [8], to facilitate the experience of inhabiting a conducive acoustic space. The pieces unfolded gradually, with a foundation of simple, repetitive and slow-moving harmonic structures. Complex melodies were avoided, and expectations of rapid change were minimized. However, random seeding and generative algorithms were used to produce intricate movements in the smaller details throughout the earlier stages of each piece, in order to give active attention to something to latch onto. This gradually shifted into a simpler sustained structure towards the middle of the work and returned (to a lesser extent) towards the end, to avoid disruption if the piece was played in a recurring loop.

Drawing inspiration from Erik Satie's project of *musique d'ameublement [9]*, these pieces aimed to augment acoustic space by making it more comfortable for the activities of the listeners. However, given the centrality of early childhood development in ABC Kids listen, the long-form broadcast (or on-demand) without interruption also provided an opportunity to introduce the young listener to complex natural soundscapes and textural variation. These were included, working with the hypothesis that—particularly in hypnogogic states—exposure to these soundscapes may enrich early relationship to sound [10].

In the title for each piece, the short accompanying text on the website (written by the composer) and the sounds used, these pieces drew on ecological (relationship between humans and the environment) rather than cultural associations (human interactions), with spoken framing and associations with music found in waking life deliberately avoided. This was based on the intention that these pieces could be inhabited as unique acoustic spaces associated with sleep and relaxation. Some pieces (particularly 'Home') used aesthetic choices based on an understanding of the acoustics of pre-birth experience, in order to encourage associations with safety and closeness.

Of course, the intentions of the composer become largely irrelevant once the works are disseminated, unless there is a reciprocal loop between composer and listener as co-creators in the production of the listening experience. Susini, Houix and Misdariis [11] propose a relationship between sound creation and research in auditory perception wherein sound design is informed by perception research through a 3-step iterative process of analysing, creating and testing. This occurs through the co-contribution of stakeholders: 'researchers', 'composers' and 'users'. The current study is useful insofar as it serves to explore these compositional premises through a survey into audience reception. By doing so, it opens the ongoing development of the *Sleep Through* series into a wider conversation between stakeholders.

3. METHOD

A qualitative survey was applied to explore how Sleep through was used, and to gather information about its efficacy from the child's parents and/or caregivers.

3.1 Survey design

An online survey using Qualtrics (www.qualtrics.com) was developed to collect data for this research. There were 8 questions in total, with the first 3 being open-ended questions as to why participants choose to listen *Sleep Through* music, and were subjected to thematic analysis. Questions 4 to 7 investigated what makes these nature-based soundscapes effective or otherwise in supporting sleep. The wording of the questions are shown in Table 1:

	Question text / response options
Q1 Open-ended question	Under what circumstances are you or your child listening?
Q2 Open-ended question	What are your aims in listening to *Sleep Through*, and what works and what doesn't work for you or your child?
Q3 Open-ended question	Which compositions in particular did you use/listen to the most and why?
Q4 Multiple choice	What musical aspect did you think was effective listening to the *Sleep Through* compositions for you or your child? • Use of low-pitched sounds. • Use of high-pitched sounds. • Use of white noise. • Use of nature sounds. • Use of regular beating/pulsing. • Use of irregular beating/pulsing. [see Figure 2 for answer options.]
Q5 Rating Scale	How often have you or your child listened to or used *Sleep Through*?
Q6 Rating Scale	On average, for how long do you or your child listen to *Sleep Through* compositions?
Q7 Rating Scale	How effective were the compositions to you or your child? Please select the best answer possible.
Q8 Open-ended question	If you would like to add any further information, please do so here.

Table 1. Survey question response types and wording

3.2 Stimuli (*Sleep Through*)

As indicated in section 2, *Sleep Through* is a program on *ABC Kids Listen* (https://www.abc.net.au/kidslisten/) that is designed to help infants *Sleep Through*out the night. At the time of the study, there were 18 different soundscapes, all of which were composed by author AH.

The *Sleep Through* soundscape tracks are laid out in blocks, with an image and a short description shown corresponding to the title of the soundscape track (see Figure 1). Participants chose their soundscapes, and streamed or download the composition in their own time.

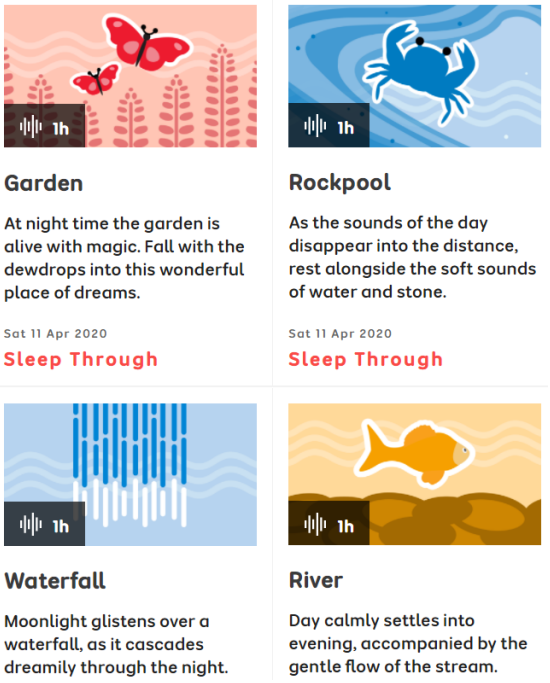


Figure 1. Hulbert's (2020) *Sleep Through* soundscape layout in: https://www.abc.net.au/kidslisten/sleep-through/. Note that the list shown in the Figure is a sample of the available soundscapes. See Figure 4 for a longer listing of the soundscape titles.

3.3 Participants and Procedure

Participants were assumed to have listened to the ABC Kids listen *Sleep Through* soundscapes prior to taking the survey. A survey link was distributed through social media; gaining attention from the ABC Kids followers (either for the child or their care-givers). Participants were able to listen to the stimuli for as long as they wanted. Furthermore, participants needed to provide consent for their participation in the survey and those wanting to discontinue had the right to withdraw at any time. After two months of gathering participants, the survey was brought to a halt. Forty-four participants commenced the survey, with 21 participants completing the entire survey. The analysis of results examines completed responses only.

4. SURVEY RESULTS AND DISCUSSION

4.1 General Usage and Effectiveness of *Sleep Through*

76% of the participants used *Sleep Through* more than 8 times, with 52% of those using it for more than an hour and/or on repeat. All participants gave one of the two highest ratings regarding the effectiveness of *Sleep Through* (Question 7): 60% found *Sleep Through* to be quite effective and 40% of people found *Sleep Through* to be very effective. In terms of the effectiveness of previously investigated musical characteristics regarding to Question 4, [12-14] Figure 2 shows the graph of the results. Here participants most frequently reported low-pitched and nature sounds as effective characteristics of *Sleep Through*, with high pitched and irregular beats least effective. This is consistent with the analysis of music characteristics used to aid sleep by young adults [15] and is discussed further in section 4.2.5.

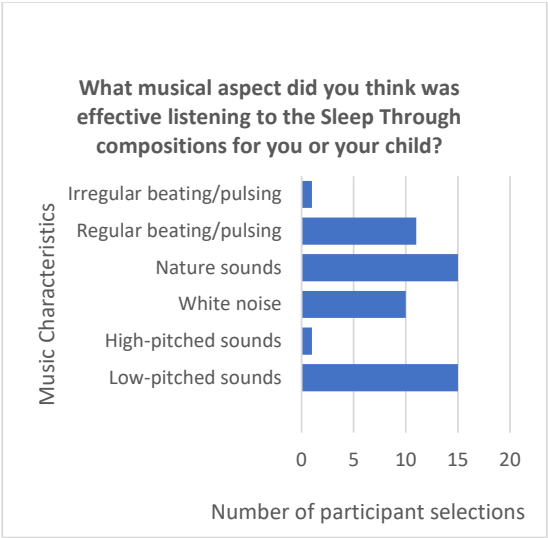


Figure 2. Distribution of answers to Question 4. What musical aspect did you think was effective listening to the *Sleep Through* compositions for you or your child?

4.2 Themes

NVivo, a qualitative data analysis software was used to iad with the organization of a thematic analysis of participant responses taken from the Qualtrics survey in categorizing certain key words or synonyms to appropriate themes. Several ways of organising the data according to theme groups were considered, and were presented as theme maps. We discuss the two most convincing theme maps. Themes were identified using a directed content analysis approach [16].

Several themes identified were based on research by Dickson and Schubert [7], Mazzarolo [17] and Trahan, et al. [18] on why music was effective in assisting sleep. Dickson and Schubert [7] found that relaxation, distraction, entrainment, masking, enjoyment and expectations were important themes. Their findings sifted through 101

publications in relation to music and sleep via online database searches and searching references. Mazzarolo [17] expanded Rogers [19] research, and noted that *Sleep Through* was based on predetermined principles (see section 2). Similarly, Trahan, et al. [18] grouped their findings using 4 main, overlapping themes: Distract, Provide, Habit and State. By applying the amalgamated themes to the survey data and then refining, the following themes were identified: Relaxation, Association, Distraction and Auditory Masking is shown in Figure 3. The possible addition of the theme Habit was also discussed (see section 4.2.3). These themes help to understand the different ways in which the compositions benefit the listener.

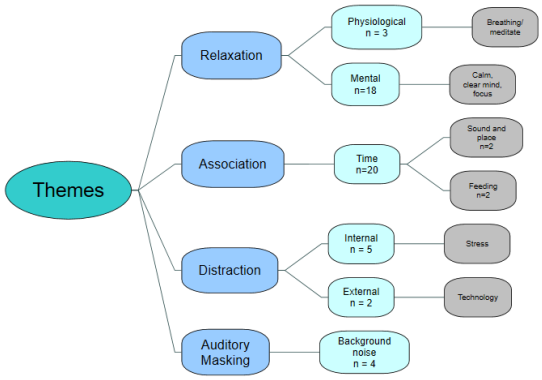


Figure 3. Theme map from the *Sleep Through* survey. The first branch of categories shows the main uses of *Sleep Through*. The second branch of categories are subthemes and the third branch are prominent responses from the subthemes identified in the survey. 'n' is the number of participants whose responses could be coded into the subtheme. The sum of n is greater than the total N (= 21) because responses from a participant could be coded into more than one theme.

4.2.1 Relaxation

Participants responded to the slow, ambient sounds of the *Sleep Through* playlist as a major source of relaxation. Relaxation refers to the tension released from the body (physiologically) and mind (mentally) to minimise and combat stress and anxiety [18]. Overall, out of 21 participants who indicated terms such as 'relax' or 'calm', 18 of them were closely linked to a mental state of mind. Examples included "it helps me relax into sleep especially after a hectic day" (P15) (P = participant code), "to calm down and clear my head (P13)" and "I find it makes me feel a lot calmer and sleepier" (P39). This theme is therefore related to the distraction theme we discuss below. 20% of these participants used music for sleep in connection with the physical process of deep breathing whilst the remaining participants used music for clearing and slowing the mind before bedtime.

The physiological process of deep breathing can be relaxing because relaxed breathing patterns calms the autonomic nervous system [20]. Nanthakwang, et al. [21] investigated deep breathing exercises and body scan meditation combined with sedative music, finding improvement in sleep quality of adults when sedative music was playing. This is in line with the present study where three participants reported effects that were linked to physiological aspects of relaxation. P2 stated, "The rising and falling of the sound is good to regulate my breathing" and P5 revealed that "Tools such as the ABC Listen app have helped us build their skills in body awareness, step by step relaxation, and deep breathing" (Author note: The ABC Listen app can also be accessed via the Kids listen content). Additionally, P10 used *Sleep Through* for their children to "meditate before bed to get them falling asleep quicker".

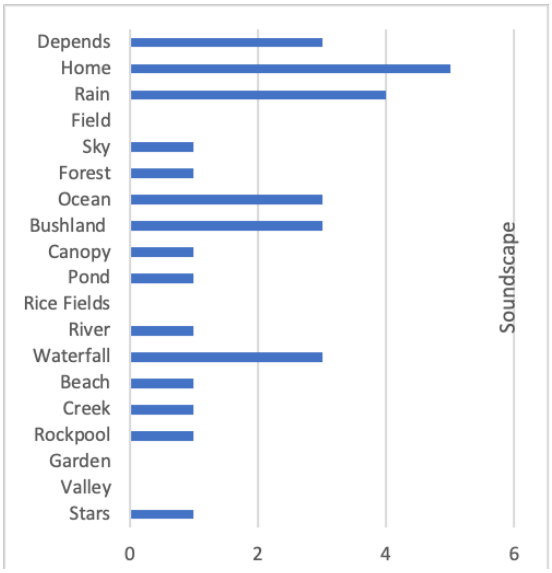


Figure 4. Frequency of selection of preferred *Sleep Through* soundscapes listened to before or whilst in bed. Note that the 'Depends' option reflects participants who selected more than one *Sleep Through* soundscape.

4.2.2 Association

Association refers to a mental connection between concepts, mental states or events, that usually stems from specific experiences [22]. We ascertained the significance of association from responses to Question 3 where participants chose their favourite soundscapes along with the reason for its choice. Figure 4 shows that the majority favoured the soundscape named 'Home' and the reasons for liking this soundscape were based on the composition itself and the description provided in the stimuli. The description of the soundscape reads "Warmth, love and peace fill this special space. The calming sound of a heartbeat and distant music keep you company as you drift into slumber." Five participants specifically mentioned that they use the 'Home' soundscape to help bring themselves or their child to sleep. The composition of 'Home' could be described as containing elements that are monotonous and muffled, where layers of audio are sounding simultaneously. There is a regular drone of harmonies that come and go in waves. Each wave gradually crescendos and decrescendos, with a period of about 10 seconds. Underneath the layer of waves, a pulsing beat can be heard throughout

the track (in particular, at 33 minutes into the track), evoking the sounds of a heartbeat inside the mother's womb. In between hearing the sound of waves and heartbeat, modulated broadband noise mimics sound transmitted through amniotic fluid. As all three layers are combined, care-givers use the track for both themselves and their child in a manner that resembles shutting down the mind and body to rest, just like an unborn baby is sheltered from the outside world while resting in their mother's womb [17]. In particular, P12 mentioned the soundscape "Home", as being most calming, saying that "the subtle heartbeat sounds help calm my son to sleep". This particular participant enjoyed playing 'Home' for his son, and reported a decrease in anxiety and undisturbed sleep. Thus, it is not a type of stimuli that would arouse the listener, but rather create a comforting environment to support sleep, and its potential to support sleep in infants was espoused by Mazzarolo [17].

The incongruence between sound and place can remind individuals of times when they were connected to nature [23]. Even though 'Home' was the single most favoured soundtrack (Figure 4), the majority of selected soundscapes were water-based (i.e. 'Rain', 'Ocean' and 'Waterfall'). This was attributed to the participants having been to a place where they have seen rain, an ocean or a waterfall, being suggestive of the Association theme. As a further example of this association between sound and place, P5 comments "If we have been to the beach that day, they always pick ocean". P5 plays the soundscape 'Ocean' for their kids throughout the night as a reminder of the past events they had enjoyed or would want to relive the experience. P40 listened to 'Stars' whilst watching the stars before sleep in order to calm down to "help with faster sleeping". Both these participants have associated music with activities performed earlier.

4.2.3 Habit formation and non-sleep related reasons

We were not unanimous in choosing Habit as a theme into which responses could be placed, because those responses generally overlapped with other themes (hence not shown in Figure 3). However, habit formation has in recent years been proposed as possible explanation why playing music each night is a successful way of aiding sleep [18, 24]. In the present study, most participants played *Sleep Through* as an indication that it is time to go to bed, with explicit example such as "I use it to get to sleep" (P7), "to assist with falling asleep" (P4) and "to assist my 2year old in settling for sleep – she seems to have a good association between the music and sleep" (P6). Habit was included in an alternative version of our thematic analysis. We decided to omit Habit as a theme in our final analysis because of its considerable overlap with the Relaxation theme and because some habit related responses could be incorporated into the Association theme. With the alternate analysis, the theme Association linked with the subtheme 'sound and place', and Time related references became as subtheme of Habit signalling bedtime: "I use it to get to bed", "assist with falling asleep" etc. Furthermore, the responses related to breastfeeding also fitted comfortably as a subtheme of Habit formation. For the interest of the reader, Figure 5 shows this alternate theme map that includes Habit.

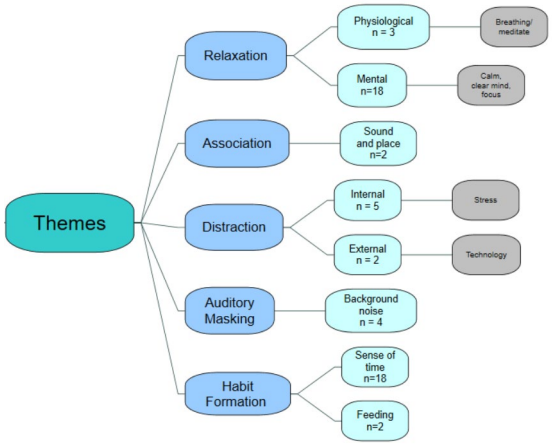


Figure 5. Alternate theme map from the *Sleep Through* survey, with the Habit theme included. Notice the large number of overlapping themes. See Figure 3 for more details.

4.2.4 Distraction

Sleep Through was beneficial because engaging in the soundscapes diverted the listener from focusing on stressful thoughts. Five participants in the present study expressed that *Sleep Through* helped reduce distractions from their own thoughts, as shown in Table 2. Notice the considerable overlap between this theme and the Relaxation theme discussed above. Indeed, it may be possible that relaxation mediates sleep, and distraction is a means to facilitate relaxation, but in and of itself does not offer a direct psychological pathway to sleep [for further discussion see 25].

Participant	Response classified as Distraction theme
P15	"I find 'Rain', 'Forest' and 'Bushland' most relaxing and the repeat of sounds helps me to focus on sound rather than running over things in my head".
P17	Plays all the nature soundscapes for her daughter to "stop and slow down to go to sleep"
P5	Uses *Sleep Through* to help switch off at night – "my son is always thinking and my daughter fidgets a lot"
P40	Plays *Sleep Through* to "help destress and help with faster sleeping"
P41	Uses *Sleep Through* "for my brain to slow down".

Table 2. Response classified of Distraction theme

Not only do internal distractions contribute to the interference of focus, but external distractions are also factors that involve visual triggers and online social interactions via the use of technology. From the survey, 2 participants reported using *Sleep Through* as a way to fix the previous bad habits when getting into sleep for healthier sleep habits. Notice again the overlap between themes, this time between distraction and the possible theme of habit. P16 uses *Sleep Through* to "avoid watching movies while he falls

asleep" and P39 plays *Sleep Through* in bed to limit the use of being on the phone scrolling through social media. Both of these participants wanted to reduce the amount of time spent on technology that could potentially tire the eyes. Brockmann, et al. [26] mentions evening exposure from watching television is associated with poor quality sleep in preschool children, due to the constant light exposure in a dark room. Long-term blue light exposure via phones, television and computer screens can damage the photoreceptors in the eye [27]. Hence, the option for having soundscapes running in the background is a more subtle approach in getting the child to bed.

4.2.5 Auditory Masking

Auditory masking drowns out or minimizes unwanted background noise enabling focus on the sounds we want to hear and can be applied to music to aid sleep [7]. The unwanted sounds can be intermittent sounds filling our surrounding environment, causing a disturbance for both the individual and surrounding people [7]. Hence, the hustle and bustle of unwanted sounds during sleep can cause poor sleep quality and quantity. Three participants reported using *Sleep Through* for their child and themselves before bed by leaving the playlist running throughout the night with P8 saying "I use it for the toddler as background sound to sleep and aid uninterrupted sleep". P6 says "It assists in blocking background noise in the house" and P12 enjoys the "calm sounds that play softly in the background and help drown out sounds of the rest of the house." Likewise, Xie, et al. [28] investigated the influence of ocean sounds on sleep patterns in an intensive care unit. Patients who were grouped to a condition for receiving ocean sounds (based on white noise stimulus) reported higher scores in the quality and quantity of sleep than those who had to slept with no music or sound. This was explained in terms of patients exposed to ocean sounds feeling a 'low' level of arousal, putting them in a state of calmness resulting in better sleep. However, there can be sounds in the sleep-aiding stimulus which may inadvertently interfere with sleep resulting in participants being woken up. P12 continued "I always wake up to the sky soundscape. The bing and bong noises are too high pitched." As our brain is constantly active in cycling through REM and NREM stages of sleep, sudden high pitched, loud sounds may result in the disruption of sleep, providing ineffective masking of environmental sounds [15]. In this case, the occasional high-pitched sounds reported by P12 was more overbearing than the unwanted noise, resulting in P12 waking up in the middle of the night.

4.2.6 Other uses of Sleep Through

Even though *Sleep Through* was aimed to induce sleep, adults reported using these soundscapes for non-sleep related applications. P39 said, "When I'm listening to *Sleep Through* whilst studying, I find myself concentrating more and being in the zone." Although not the specific aim of the soundscapes, the response should not be that surprising, with evidence existing, for example in research by Newbold, et al. [23] that auditory stimulation can help maintain attention and concentration.

Also not directly related to supporting sleep was the reported use of the soundscapes for breast feeding. P3 was explicit in reporting benefits to both sleep and breast feeding, using *Sleep Through* "To sleep calmly and soundly than before. Also, for my nearly 2-year-old to settle straight back to sleep after her breastfeed together". Similarly, P8 used *Sleep Through* "for a small baby when up feeding throughout the night to aid uninterrupted sleep and help join sleep cycles." Music used in breast feeding enhances the attachment between the mother with her infant [29].

5. CONCLUSION

This paper investigated audience responses to the soundscapes composed as part of the ABC commissioned series *Sleep Through* composed by author AH. When prompted for the reasons that the compositions were selected, and to assess their efficacy, an overwhelming number of responses were consistent with established views on how music is used to aid sleep for adults, with key themes being Relaxation, Association, Auditory Masking, Distraction, and Habit. Relaxation was most frequently reported, and some overlap between relaxation and distraction were also observed. The 'Home' soundscape was the most frequently selected by the sample, possibly because of its association with events in the day, or because it applied sounds that mimic gestational womb sounds, triggering the comfortable past for the infant. The water-based connection as an explanation of the soundscapes is also reflected in the frequent choice of water-related soundscapes by the participants, although this too was frequently related to activities that took place during the day, such as going to the beach. The proposed themes while distinct, showed numerous cases of overlap.

The compositions were also considered highly useful in helping young families commence and maintain healthy sleeping behaviours, but interestingly other, non-sleep benefits were observed, including using the soundscapes for breast feeding, or purely for pleasure.

The near absence of criticism of the works is encouraging in identifying alignment of the initial propositions with the reception of the works, keeping in mind that respondents were drawn from the ABC's social media channels and therefore more likely to have self-selected according to positive associations with the brand. Some of the works surveyed were composed during and after research from Mazzarolo [14], speaking to the growing dialog between the community, composer and empirical researchers surrounding this unique broadcast. The one finding that contradicted initial compositional premises (namely the impact of short, high sounds in one piece) affords a valuable insight for future compositions. Also interesting is the close alignment between the empirically arrived at themes and three of the four foundational approaches reported by the composer: Auditory Masking; Relaxation through Entrainment, and; Association and Familiarity.

The study is obviously limited, in that the participation rate was small, due in part to the ethics requirements. The ethics requirements required detailed explanations. Since participants were given the option of identifying them-

selves, they needed to be aware of this before commencing, if they agreed to participate. This was a factor in dissuading a larger sample from participating. The quantitative results were therefore presented in descriptive form only, but those data and the examination of the open-ended responses still produced responses consistent with previous research, which was largely based on adult uses of music for aiding sleep.

The current study, for obvious reasons, relies upon adults reporting infant experiences, and future research will be needed to determine what special aspects of music and soundscapes might need to be adapted to better serve infant sleep. But our findings are consistent with the theoretical position of the kind of music that might be suitable for infants as proposed by [17], as well as the literature highlighting musical characteristics most preferred by adults to facilitate sleep [15]. While larger-scale studies await, our findings suggest that there are some commonalities in the approaches to sound throughout the *Sleep Through* series that support sleep, and that these can be further developed to support the positive health outcomes that result from a good night's sleep for both children and adults.

Acknowledgments

We wish to thank the Australian Broadcasting Corporation (abc.net.au) for their invaluable support of this project. The paper is based in part on author LW's Honours thesis.

6. REFERENCES

[1] A. Kozbelt, "A quantitative analysis of Beethoven as self-critic: implications for psychological theories of musical creativity," *Psychology of music.,* vol. 35, no. 1, pp. 144-168, 2007, doi: 10.1177/0305735607068892.

[2] D. J. Levitin and L. L. Cuddy, "Editorial: Introduction to the Angel of Death Project," *Music Perception: An Interdisciplinary Journal,* vol. 22, no. 2, pp. 167-170, 2004, doi: 10.1525/mp.2004.22.2.167.

[3] R. Reynolds, "Compositional Strategies in The Angel of Death for Piano, Chamber Orchestra, and Computer-Processed Sound," *Music Perception: An Interdisciplinary Journal,* vol. 22, no. 2, pp. 173-205, 2004, doi: 10.1525/mp.2004.22.2.173.

[4] G. G. Alvarez and N. T. Ayas, "The impact of daily sleep duration on health: a review of the literature," *Progress in cardiovascular nursing,* vol. 19, no. 2, pp. 56-59, 2004, doi: 10.1111/j.0889-7204.2004.02422.x.

[5] E. K. Tham, N. Schneider, and B. F. Broekman, "Infant sleep and its relation with cognition and growth: a narrative review," (in eng), *Nat Sci Sleep,* vol. 9, pp. 135-149, 2017, doi: 10.2147/NSS.S125992.

[6] E. Costa-Giomi, "Infant home soundscapes: A case study of 11-month-old twins," in *International Perspectives on Research in Music Education*, London, UK, G. a. Boal-Palheiros, Ed., 18-22 July 2016, pp. 70-78.

[7] G. T. Dickson and E. Schubert, "How does music aid sleep? literature review," *Sleep Medicine,* vol. 63, pp. 142-150, 2019/11/01/ 2019, doi: https://doi.org/10.1016/j.sleep.2019.05.016.

[8] W. W. Gaver, "How do we hear in the world? Explorations in ecological acoustics," *Ecological psychology,* vol. 5, no. 4, pp. 285-313, 1993.

[9] N. Bernardini, "Erik Satie's Musique d'Ameublement, some ninety years later," 2008, pp. 1-12.

[10] L. F. Gooding, "Using music therapy protocols in the treatment of premature infants: An introduction to current practices," *The Arts in Psychotherapy,* vol. 37, no. 3, pp. 211-214, 2010/07/01/ 2010, doi: https://doi.org/10.1016/j.aip.2010.04.003.

[11] P. Susini, O. Houix, and N. Misdariis, "Sound design: an applied, experimental framework to study the perception of everyday sounds," *The New Soundtrack,* vol. 4, no. 2, pp. 103-121, 2014.

[12] S. Ogata, "Human Eeg Responses to Classical Music and Simulated White Noise: Effects of a Musical Loudness Component on Consciousness," *Perceptual and motor skills.,* vol. 80, no. 3, pp. 779-790, 1995, doi: 10.2466/pms.1995.80.3.779.

[13] Y. Wang, A. Rivard, C. Guptill, C. Boliek, and C. Brown, "Characteristics of sleep-conducive music: A narrative evidence review.," *Advances in Social Sciences Research Journal,* vol. 7, no. 3, pp. 430-437, 2020, doi: 10.14738/assrj.73.7968.

[14] K. S. Yeo, "Acoustic characteristics of the forest sounds inducing sleep," *Information,* vol. 18, no. 10, p. 4407, 2015.

[15] G. T. Dickson and E. Schubert, "Musical Features that Aid Sleep," *Musicae Scientiae,* p. 1029864920972161, 2020, doi: 10.1177/1029864920972161.

[16] H.-F. Hsieh and S. E. Shannon, "Three Approaches to Qualitative Content Analysis," *Qualitative health research.,* vol. 15, no. 9, pp. 1277-1288, 2005, doi: 10.1177/1049732305276687.

[17] I. Mazzarolo, "Infant Sleep through Noise and Music," Empirical Musicology Laboratory, UNSW, Sydney, Australia, 2019. [Online]. Available: https://figshare.com/articles/Infant_Sleep_through_Noise_and_Music/7762739

[18] T. Trahan, S. J. Durrant, D. Mullensiefen, and V. J. Williamson, "The music that helps people sleep

and the reasons they believe it works: A mixed methods analysis of online survey reports," *PLoS One,* vol. 13, no. 11, p. e0206531, 2018, doi: 10.1371/journal.pone.0206531.

[19] K. Rogers. "Six new ways to get a good night's sleep." https://www.abc.net.au/kidslisten/ideas/earlylearning/new-episodes-added-to-sleep-through/10766646 (accessed 31/08/2020, 2020).

[20] L. Kuula *et al.*, "The Effects of Presleep Slow Breathing and Music Listening on Polysomnographic Sleep Measures - a pilot trial," *Scientific Reports,* vol. 10, no. 1, 2020, doi: 10.1038/s41598-020-64218-7.

[21] N. Nanthakwang, P. Siviroj, A. Matanasarawoot, R. Sapbamrer, P. Lerttrakarnnon, and R. Awiphan, "Effectiveness of Deep Breathing and Body Scan Meditation Combined with Music to Improve Sleep Quality and Quality of Life in Older Adults," *The Open Public Health Journal,* vol. 13, pp. 232-239, 2020, doi: 10.2174/1874944502013010232.

[22] S. B. Klein, *Learning: Principles and applications*. Sage Publications, 2018.

[23] J. W. Newbold, J. Luton, A. L. Cox, and S. J. Gould, "Using nature-based soundscapes to support task performance and mood," in *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, 2017, pp. 2802-2809.

[24] G. T. Dickson and E. Schubert, "Music on Prescription to Aid Sleep Quality: A Literature Review," *Frontiers in Psychology,* vol. 11, 2020.

[25] G. T. Dickson and E. Schubert, "How Does Music Aid Sleep? Literature Review," *Sleep Medicine,* 2019.

[26] P. E. Brockmann, B. Diaz, F. Damiani, L. Villarroel, F. Núñez, and O. Bruni, "Impact of television on the quality of sleep in preschool children," *Sleep Medicine,* vol. 20, pp. 140-144, 2016, doi: https://doi.org/10.1016/j.sleep.2015.06.005.

[27] G. Tosini, I. Ferguson, and K. Tsubota, "Effects of blue light on the circadian system and eye physiology," (in eng), *Mol Vis,* vol. 22, pp. 61-72, 2016. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4734149/.

[28] H. Xie, J. Kang, and G. H. Mills, "Clinical review: The impact of noise on patients' sleep and the effectiveness of noise reduction strategies in intensive care units," *Critical Care,* vol. 13, no. 2, p. 208, 2009, doi: 10.1186/cc7154.

[29] D. E. Procelli, "The Effects of Music Therapy and Relaxation Prior to Breastfeeding on the Anxiety of New Mothers and the Behavior State of Their Infants during Feeding," Master of Music, Music, Florida State University, West Palm Beach, Florida, 2005.

DESIGN CONSIDERATIONS FOR SHORT ALERTS AND NOTIFICATION SOUNDS IN A RETAIL ENVIRONMENT

Gustav F. ARFVIDSSON (garf@kth.se) [1], **Martin L. ERIKSSON** (martin@soundmark.se) (0000-0002-7951-2089) [2], **Håkan LIDBO** (hakan@hakanlidbo.com) [3], and **Kjetil FALKENBERG** (kjetil@kth.se) (0000-0003-4259-484X) [1]

[1] *School of Electrical Engineering and Computer Science*, **KTH Royal Institute of Technology**, Stockholm, Sweden
[2] *Media and Design*, **University West**, Trollhättan, Sweden
[3] **Rumtiden**, Stockholm, Sweden

ABSTRACT

The design and noticeability of alert sounds have been widely researched and reported, and not least, notification sounds are ubiquitous in both software and hardware product development. In an ongoing research project concerning the retail industry, we aim at designing short alert sounds that only grab attention from one group of customers, while others do not register the alerts: this particular aspect has to our knowledge not yet been studied. To establish design guidelines for such alert sounds, we conducted an experiment where test subjects would experience ordinary shopping activity including background music and an ambient soundscape in a virtual reality clothing store, but with added alert sounds. We tested, specifically, six differently designed sound alerts belonging to two classes: contextual-specific congruent sounds, and incongruent sounds that did not fit the sonic context. The results disproved our assumptions that incongruent sounds would outperform the congruent and thus in the context more anticipated sounds. The findings suggest that alert sounds can be designed with subtlety and still be noticeable and that customers will not necessarily be annoyed. We present here a first approach towards design guidelines for short alert sounds in a shop environment.

1. INTRODUCTION

Notification sounds are omnipresent in our lives: we get exposed to alert sounds for communication, interaction, status, and safety situations, and we get these from our mobile devices and computers, vehicles, household machines, and public buildings. Furthermore, the alert sound designs span from clicks and beeps to human voices and music excerpts.

In an ongoing research project "Sonification of store goods" involving retail, theft, and shopping, we need to reconsider attentiveness towards notification sounds: we aim at designing short alert sounds for non-critical contexts that only grab attention from targeted actors in an environment. The main aim for our project is to discourage and prevent shoplifting by playing alerts as sonifications of interactions with goods in shops without discomforting regular customers or distracting employees. Ideally, the sonifications should not attract attention from others than store clerks and shoplifters, and the sounds should not reduce the overall shopping experience.

Using sonification for monitoring state can free up cognitive resources [1], cutting back costs on expensive video surveillance systems, and open for live monitoring where information on what goes on in the store can be conveyed to the staff in real time. It also solves the ethical question of storing customer information in the form of video material, with reduced impact on personal integrity. However, not much has been done in terms of using sonification in store environments.

We have identified knowledge gaps in several aspects of this particular challenge, for instance, how fast do listeners react to sounds (with head movements), how do sounds that have either context-specific congruence or incongruence to the store's sonic ambience differ in terms of grabbing attention, do sounds with early onsets perform better for notification and localization than slow onsets, and will repeated exposure increase or decrease attention.

For this present study, our goal is to investigate the effects that exposure to different sound types have on customers and clerks in a virtual store. The research question is to find if visitors to a (virtual) store will have their attention drawn towards sound alerts being played depending on the type of sound. Our assumption was that congruent sounds would draw less attention than incongruent sounds which diverge from the sonic store environment. Especially, we expected a recording of chirping birds and a metallic wind chimes to be overrepresented in terms of detection as these sounds were chosen intentionally to be detached from the context.

In the next section, we present the contextual framework of the project, namely loss prevention in retail, and also necessary theory on perception, sonification, sound design, and the experimental environment. The method section describes both the sound design and the practical test design in a semi-controlled experimental setting. The result section focuses to some extent on head movement data, while in the discussion we interpret the results from a practical sound design perspective. The paper concludes with a first approach towards alert sound design considerations and implications in a non-critical context.

2. BACKGROUND

There is a documented need for surveillance in stores. The total loss due to shoplifting has reached US$10 billion yearly in the United States [2]. To counter this, stores adopt different methods of surveillance such as increasing monitoring staff, security guards and cameras, and electronic alarm systems such as electronic article surveillance and RFID tags; the most effective countermeasures generally involve human factors [3].

In addition to anti-theft alarms in stores using RFID tags and alarm noises when someone walks out of the store, using loud sounds to prevent, disrupt, or rectify undesired situations are widely implemented in the society today. Car alarms, for example, typically appropriate the car horn; however, there are also suggestions for alternative designs to be found in the literature, such as musically informed car alarms [4]. Another common case is in hospital environments where doctors are constantly exposed to a great number of different alerts and warnings from apparatus, often similarly-sounding, with the result that alarms are missed, ignored, or even turned off [5].

2.1 Perception and Localization of Alert Sounds

The perception of sound is a heavily researched area and has been much influenced by Lord Rayleigh's "duplex theory" of sound localization as a combination of interaural differences from sound pressure level and phase [6]. Building on his observations, studies of localization of sound in the horizontal plane have found that, although stimuli could be located with reasonable accuracy, test subjects confuse stimuli presented in front and from back. Specifically, most errors occur around 3000 Hz and decline at higher or lower frequencies [6].

Directional hearing is the ability to locate the position of a sound source. This ability depends on comparisons between the acoustic inputs from the two ears, while pitch and intensity can be derived from only one ear [7]. The direction is determined binaurally from the time difference and the loudness difference of the sound waves reaching the two ears. The onset and beginning part of a sound is more important for our perception than later parts of a sound [8].

In a study on localization of sound in rooms, Rakerd and Hartmann found that impulsive tones with short onset and offset were more accurately located than those with slow onset due to the precedence effect; also, tones of longer duration that gave no precedence effect showed large individual differences [9]. There were no measurable effects on pulse durations ranging from 5–2000 milliseconds, instead Rakerd and Hartmann proposed the "plausibility hypothesis" where listeners ignore ongoing location cues after the onset when these are implausible.

2.2 Perceptual Attention and Urgency

Perceptual attention is defined as the ability to extract relevant information from complex surroundings that cueing, i.e., playing a sound or a warning about a sound before the actual sound is played, can improve our ability to detect and to locate a sound [10]. Therefore, the use of the same sound, or a sound the one is used to hear, would increase the ability to detect and locate it. However, with simultaneously played sounds there is a risk of a performance reduction in terms of reaction time [11], and with repeated sounds there is a risk of increased annoyance [12].

Alert sounds typically have been designed and implemented to communicate urgency and attract attention towards critical events that require immediate action [13]. It is reasonable to say that attentiveness to urgent notification sounds has been widely researched and reported; one example is the concept of "attensons" as put forward by Hellier and Edworthy, which are attention sounds designed from established perceptual and psychophysical principles such as signal-to-noise ratios [14].

The auditory system is built to process simultaneous and overlapping stimuli, although dependent on attention [15]. One study showed that people working in security operations centers with computer-network security were aided by sonification in ways that enabled peripheral monitoring in busy multitasking environments [16]. Sound alerts also has the benefit of utilizing the fastest of the human senses [17].

Our perceptual and cognitive knowledge of an environment is grounded in our ability to learn from previous experiences. We use this knowledge in relation to a context to predict which sounds that are likely to appear, but also to reject interpretations incongruous with a context [18]. When audiovisual sensory information is unrelated it leads to an uncertainty of interpretation, causing an attentional focus on identifying what is incongruent [19].

2.3 Sound Design and Contextual Sounds

Notifications can be realized with sonification: here we define sonification as systematically translating sensor data to non-vocal sounds. In particular, we use an event-based approach for monitoring state in a multimodal environment [1], by playing sound recordings to describe the interaction taking place.

There is a need for design of more aesthetically pleasing sonification designs and alert sounds [20]. Several authors have also stressed the importance of designing alert sounds with a high level of ecologic validity [21, 22] to match the function. In this study, sounds that would grab attention but not be disturbing were applied in a virtual store. This type of sound design where sounds that should be heard without being too cognitively demanding has been approached in a previous study [23].

Ecological validity in terms of good recordings or realistic sound simulations are not necessarily the most efficient design strategy to convey information. Instead, context-specific congruent low-level models that sacrifice realism for plainness have proven to be effective in communication, as shown in the research literature on sound objects and cartoonification [24]. Our assumption for the study was that incongruent sounds unrelated to the actions and the environment would inevitably draw more attention than the congruent sounds, and would also be perceived as more disturbing.

In the present work, we are not designing alerts or sonifications that communicate immediacy or urgency, but arguably with more coherence than the range of less urgent notifications that has resulted from the growing all-purpose use of smartphones and other technologies.

2.4 Virtual Reality Environment

Virtual reality (VR) is a computer generated interface that realistically simulates a physical environment. It is typically experienced through a head-mounted display (HMD) such as the Oculus Quest [1]. There are many advantages with using VR in research studies, such as increased experimental control, isolating test variables, and of approaching multiple variables in controlled conditions. Commercially available VR products like the Oculus Quest makes it possible to track body motion and head movements, which in many circumstances facilitate running complex experiments which would otherwise be difficult [25].

The kind of VR used in this study, HDM, is defined by providing 3D stereo vision, surround vision and user dynamic control of viewpoint [25]. In addition, sound was played uncompressed through Audio-Technica ATH-M50X stereo headphones which fit comfortably on the HDM. The spatialization mode in the software was without corrections for vertical head displacement, only horizontal movements. These features, when implemented together, provide for an immersive experience where the user is perceptually shielded from the surroundings, but where the experience matches a real world. Studies have showed that the sense of presence and immersion is generally high [26].

3. METHOD

We designed the experiment such that test subjects would experience a visit to a virtual clothing store including moving, autonomous customers, background music, recorded store ambience, and added alert sounds. The main data collected for analysis were head movements and interviews, while all audio events were variables under our control. The position and rotation of the test subjects' head movements and in-game movements were sampled and saved in the Oculus Quest HMD at 10 Hz.

The clothing store VR environment and sound programming were implemented using the game engine Unity 3D. The store measured approximately 700 m^2 and its merchandise consisted of shirts, pants, hats, backpacks and belts, among others, see Fig. 1. In addition to that, six avatars, two men and four women, would walk around in the environment and interact with the merchandise.

3.1 Experiment Design

The experiment included 16 test subjects (9 female, 7 male, age 24–53). Most of them had little to none previous experience with VR environments. The subjects were randomly assigned to one of two groups, *Knowing* and *Unknowing*, and were told they would play the part of store clerk in an informal game or VR experience where you cannot win or lose. Then, the groups were given different instructions: The knowing group was informed that alert sounds may occur in the store, which signalled that one of the avatars picked up and looked at some merchandise. They were given the instruction to experience the store and possibly pay attention to what the avatars were doing. The unknowing group was not informed about the alert sounds. They were given the instruction to experience the store and that we would conduct an interview to evaluate the "quality of the avatar's AI" without explaining what that meant.

Figure 1. The virtual store environment, which has one large space with shelves and clothing racks, one small adjacent room with more items, and one fitting room. The avatars on the platform walk around and look at items during the test.

The reason for having two conditions was to compare across the participants. In this study, the focus is on reactions to different sound types, congruent and incongruent, while attentiveness between knowing and unknowing participants are explored in more detail in a related paper [27].

To let all participants experience the store in a comparable way, they were not in control of their avatar's motion across the room, but only the head movement. In order to avoid having a strange VR sensation, the subjects were instructed to hold on to and position themselves between two chairs and follow the avatar's motion through walking and turning on spot, which through testing proved to be very helpful.

The rotation of the test subjects head movements during the test were compared to the location and time of the alert sounds, and we could see if the alerts triggered any reaction with the test subject. Data was analyzed using t-tests and Chi-square with 5% significance level. The movement data is available online. [2] The VR session lasted for 10 minutes. After the experiment, the participants were interviewed about their experience.

3.2 Sound Design

The sounds in the environment, apart from the actual alert sounds, consisted of generic background music and clothing store background noise/ambience. These sounds are also used in several associated experiments not reported here. Store ambience sound and the background music

[1] https://www.oculus.com/quest/

[2] https://annexes.smcresearch.se/2021-SMC-AELF

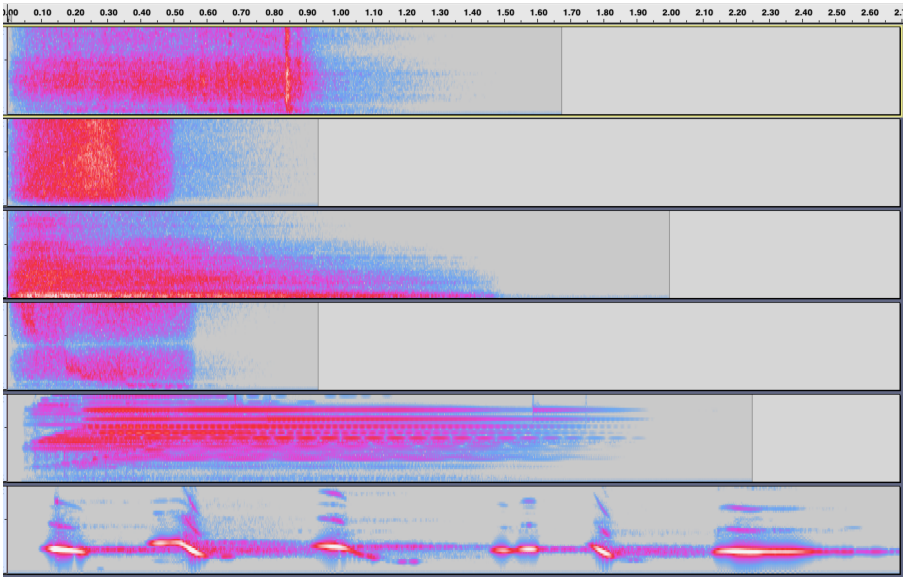


Figure 2. Spectrograms of the six alert sounds. From the top: The three congruent sounds, then the incongruent sounds sweep, chimes and bird. The sounds are also available online (see Footnote 2).

were omnipresent and played from virtual speakers placed all over the ceiling. The alert sounds were spatially separated in the VR environment, where each sound event was played from a virtual speaker close to the place of interaction. However, the alert sounds were not acoustically affected by walls and other objects. Therefore, the acoustic environment can be considered as an open space; although within the VR context, the experience is simply that of standing in a room without audible reverberation.

Six alert sounds were designed by two professional sound designers through iterations based on sound qualities such as attack, length and intensity, and did not have harmonic or tonal qualities that would conflict with the background music. The sounds belonged to one of two groups: three congruent sounds corresponded contextually with the clothing store environment and three incongruent sounds were contextually detached.

The three incongruent sounds—bird song, a time stretched sweep sound, and wind chimes—were selected on grounds of their disassociation from a store environment, and these three did not bear any internal resemblance. In particular, the wind chimes and bird song were intentionally distinctly detached from the context. See Fig. 3.2 for spectrogram representations; all sounds are also available for listening online (see Footnote 2).

The sounds considered as congruent were two recordings of a clothing hanger and a sweep-like sound, based on them mimicking the action of removing a piece of clothing from a hanger. The sweep sound was clearly resembling the hanger sounds in terms of structure and timbre, but in a cartoonified manner.

The sounds were played, in total, 25 times per test at the exact same time, position, sound level, and in the same order. Each sound had its own virtual speaker in the store,

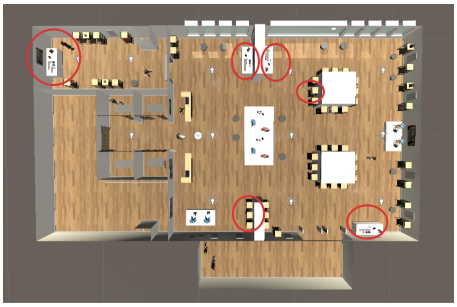


Figure 3. The layout of the virtual store environment with playing zones (virtual loudspeakers) for the sound alerts marked with red circles. Background music is played everywhere from virtual loudspeakers in the ceiling.

connected to a physical object such as a shelf, a table, or a clothing rack, see Fig. 3. The volume was set, through testing, to be just a bit louder (a few dB) than the masking sound from the background music and ambience. The perceived sound volume of the alert would however depend on the position of the virtual speaker relative to the head rotation of the avatar, and to some extent on the background music at that very moment.

Notification sounds appeared from testing to have the same level when positioned both in the middle of the room and when following the avatar along its path. While the level decreased with distance to the speaker, the distances to each of the virtual speakers were identical for all subjects as they followed a set path through the store with the set actions and movements from the avatars. Only head rotation varied between subjects. The overall sound level,

based on the background music, was set to a comfortable listening level by the subject.

4. RESULTS

First, we could confirm that head movements by the unknowing and knowing groups differ, see Fig. 4. During sound alerts, the average motion measured in angular speed and extent of rotation was almost twice as high for the knowing group, while in-between sound events being played, the amount of motion was almost the same.

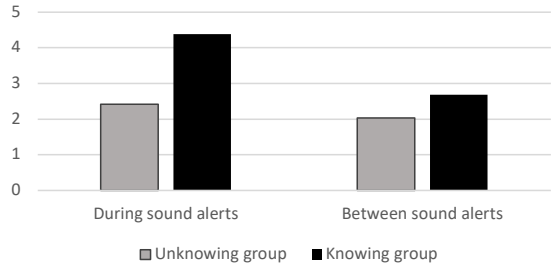


Figure 4. Head movements for the unknowing and knowing groups during sound alerts being played and in-between sound alerts, measured in speed and extent of the rotations.

The next step in the analysis of the head movement data was to determine "hits", or reactions to sounds where the head rotation pinpoints the sound source. An event would be considered a hit if the head rotation of the test subject existed in the range of 30 degrees from the direct line from the test subject to the event position. Experimenting with the angle and trying different ranges of hit area led to the conclusion that the range did not affect the number of hits considerably as the range increased, and therefore we set 30 degrees as default range.

We found that the reaction to a sound event typically came after two seconds from the start of the sound. The amount of hits would have a ceiling effect from three seconds and longer. Thus, we include hits identified between the start of the sound plus three seconds.

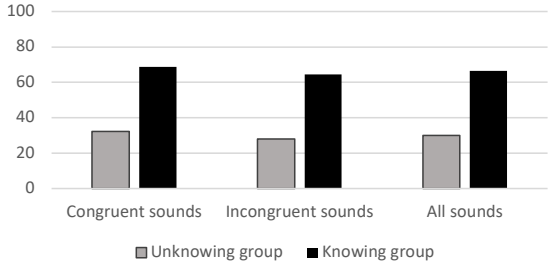


Figure 5. Reactions ("hits") for congruent, incongruent, and all alert sounds in percent for the unknowing and knowing groups.

Looking at the total amount of hits during the tests we can see that the group which were informed to look or listen for sounds outperformed the other group significantly (χ^2, $p = 0.000$), see Fig. 5. On average, the unknowing group reacted to 30% of all sounds, while the knowing reacted to 67%.

Furthermore, we found no significant effects (χ^2-tests) of the distance from the sound source to the test subject, of the physical place of the sound, of the test subject's gender, or of the sound-causing avatar's gender. The knowing group were generally unaffected by the total length of the sounds; longer sounds gave slightly more hits while medium long sounds gave fewer, but the differences were small and not significant (χ^2, $p = 0.24$). On the other hand, there was an effect of duration for the unknowing group (χ^2, $p = 0.025$) where longer sounds resulted in more hits. Sounds longer than two seconds resulted in twice as many hits on average than sounds below one second. There were no significant differences between having an early or late (100–800 ms) sound amplitude peak (χ^2, $p > 0.31$). There was no effect when the same stimulus was repeated for the unknowing group, but the knowing had a small increase of hits for a repeated sound (χ^2, $p = 0.005$). We notice a slight decline in attentiveness among the test subjects, but there are no significant effects of exposure over time.

The Bird type sound generated most hits, followed by the three congruent sounds, see Fig. 6. These differences are significant (χ^2, $p = 0.025$). The Bird sounds got most attention from the unknowing group, while the largest difference between the two groups was found for the Sweep sound. Chimes was the sound with overall least number of hits. However, these observations have not been evaluated statistically.

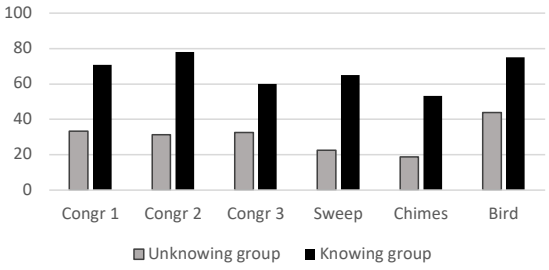


Figure 6. Reactions ("hits") to the different alert sounds in percent for the unknowing and knowing groups.

Finally, the results partly disproved assumptions that incongruent sounds would outperform congruent and more subtle sounds, see Figures 5 and 6. Instead, we notice certain patterns that will be discussed in the following.

5. DISCUSSION

Interpreting the results can give us a clue into what implications for the sound design it may result in. First off, it is possible to locate spatially separated sounds in a busy clothing store environment and directional sounds can be used. And more importantly, the sounds could be played at

a lower volume overall. It was proven that the ability to detect and locate sounds in such an environment was not an overwhelmingly difficult challenge. And furthermore, the unknowing group tended to react less to the sounds than was expected. This leads us to believe that a lower volume could be used, where those who do not listen will be even less disturbed, but those who do listen will still be able to distinguish the majority of alerts.

Another finding we did not anticipate was that shorter sounds proved to be more suited compared to longer sounds. One might think that longer sounds would have a greater impact on our perception purely because of their length and longer exposure to our ears but when it comes to raising our attention, at least in this environment, shorter sounds are more effective. Furthermore, it was shown that the onset or attack in the shorter sounds had no effect on detectability. This was also in contradiction to the hypothesis, which could be due to the short durations overall. The onset differed with a maximum of 600 milliseconds and humans typically have a reaction time to audio stimulus of 140–160 milliseconds [28].

The results show that congruent sounds generally are noticed to a greater extent than the incongruent sounds. There are differences between groups in attention towards the sound types. This can be seen particularly for chimes and bird sounds compared to the three congruent sounds. Both bird and chimes stand out from the background music and ambient sounds because of their strong and distinct harmonics, but the chimes, which we expected to be the most noticed sound of all, scored the least. This can surely be partly explained by the length of the sound, but then the sweep sound should not fall in-between.

This leads us to two somewhat contradictory conclusions. Our hypothesis that incongruent sounds stand out from the context holds in part, with different reactions from the knowing and unknowing groups. However, the congruent sounds, although more subtle, are easily and consistently noticed by the knowing group, but not by the unknowing. This might have support in the previously mentioned "plausibility hyphotesis" for localizing sounds [9].

Alert sounds for store environments *could* thus be incongruent since customers, which are represented in this test by the unknowing group, will not notice the sounds as much as the knowing group, the store personnel. However, these incongruent sounds need to be carefully designed, while congruent sounds can be implemented with less care. This encourages discreet notification designs, but also opens up the design space for shops and allows for instance freedom to develop sonic branding as part of the store's monitoring system.

The result of finding no growing sensitivity to the alerts adds to the argument that sounds could be designed in a way that is directed towards those who listen for it without them being disturbing for those who do not. These findings also suggest that alert sounds can be designed with subtlety and still be noticeable, and even that customers will not be increasingly annoyed. However, more research on how alert sounds are perceived in a real store environment in relation to pleasantness, fatigue and function over longer time is needed.

The study included only a small number of participants, which jeopardizes using statistical methods and making conclusions from these. Also, the sound stimuli design was not formally evaluated before the experiment, nor how and how often these were presented. There was no randomization of stimuli presentation. The participants did not get any training in visiting a VR environment, and the store layout was not evaluated for realism. As such, there are many uncertainties present in the study, and the results should therefore be considered as preliminary.

6. CONCLUSIONS

Based on the findings presented here, a first approach towards design guidelines for short alert sounds in a retail environment are stated as follows. Alerts can:

– be congruent with and contextually fit the environment of where they are played,
– be played at a lower volume than the background music,
– be short in length, around one second,
– be designed without much attention of attack sharpness,
– be used without concerns of growing sensitivity over time,
– be incongruent, if designed with care.

This first approach towards design guidelines will be evaluated and developed in forthcoming experiments. As such, the particular sound designs that were tested in this experiment will serve as inspiration, but should not be considered to be general design recommendations.

We believe that there are promising opportunities for sound design for marketing purposes as well as for increasing customer shopping experiences and working conditions for employees. VR was used successfully in this exploratory study where a real store would introduce a number of uncontrollable variables.

Acknowledgments

The research was funded by the Hakon Swenson Foundation and The Swedish Retail and Wholesale Council (Handelsrådet).

7. REFERENCES

[1] T. Hermann, A. Hunt, and J. G. Neuhoff, *The sonification handbook*. Logos Verlag Berlin, 2011.

[2] Y. Yamato, Y. Fukumoto, and H. Kumazaki, "Proposal of shoplifting prevention service using image analysis and ERP check," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 12, no. S1, Jun. 2017.

[3] A. Lindblom and S. Kajalo, "The use and effectiveness of formal and informal surveillance in reducing shoplifting: A survey in Sweden, Norway and Finland," *The International Review of Retail, Distribution and Consumer Research*, vol. 21, no. 2, pp. 111–128, May 2011.

[4] A. Sigman and N. Misdariis, "Alarm/will/sound: perception, characterization, acoustic modeling, and design of modified car alarms," in *ICMC - International Computer Music Conference*, Athenes, Greece, Sep. 2014. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01546060

[5] J. Frank E. Block, L. Nuutinen, and B. Ballast, "Optimization of alarms: A study on alarm limits, alarm sounds, and false alarms, intended to reduce annoyance," *Journal of Clinical Monitoring and Computing*, vol. 15, no. 2, pp. 75–83, 1999.

[6] J. C. Middlebrooks and D. M. Green, "Sound localization by human listeners," *Annual Review of Psychology*, vol. 42, no. 1, pp. 135–159, Jan. 1991.

[7] S. Kuwada and T. C. T. Yin, "Physiological studies of directional hearing," in *Proceedings in Life Sciences*. Springer US, 1987, pp. 146–176.

[8] D. Oberfeld, J. Hots, and J. L. Verhey, "Temporal weights in the perception of sound intensity: Effects of sound duration and number of temporal segments," *The Journal of the Acoustical Society of America*, vol. 143, no. 2, pp. 943–953, Feb. 2018.

[9] B. Rakerd and W. M. Hartmann, "Localization of sound in rooms, III: Onset and duration effects," *The Journal of the Acoustical Society of America*, vol. 80, no. 6, pp. 1695–1706, Dec. 1986.

[10] E. R. Hafter, A. Sarampalis, and P. Loui, "Auditory attention and filters," in *Auditory Perception of Sound Sources*, W. A. Yost, A. N. Popper, and R. R. Fay, Eds. Springer US, 2008, pp. 115–142.

[11] R. Zajdel, J. Zajdel, A. Zwolińska, J. Śmigielski, P. Beling, T. Cegliński, and D. Nowak, "The sound of a mobile phone ringing affects the complex reaction time of its owner," *Archives of Medical Science*, vol. 5, pp. 892–898, 2012.

[12] Å. Skagerstrand, S. Köbler, and S. Stenfelt, "Loudness and annoyance of disturbing sounds – perception by normal hearing subjects," *International Journal of Audiology*, vol. 56, no. 10, pp. 775–783, May 2017.

[13] E. Hellier and J. Edworthy, "On using psychophysical techniques to achieve urgency mapping in auditory warnings," *Applied Ergonomics*, vol. 30, no. 2, pp. 167–171, 1999.

[14] ——, "The design and validation of attensons for a high workload environment," in *Human Factors in Auditory Warnings*. Routledge, 1999, pp. 283–304.

[15] R. P. Carlyon, "How the brain separates sounds," *Trends in Cognitive Sciences*, vol. 8, no. 10, pp. 465–471, Oct. 2004.

[16] L. M. Axon, B. Alahmadi, J. R. C. Nurse, M. Goldsmith, and S. Creese, "Sonification in security operations centres: What do security practitioners think?" *Workshop on Usable Security (USEC) at the Network and Distributed System Security (NDSS) Symposium 2018*, Jul. 2018.

[17] J. Shelton and G. P. Kumar, "Comparison between auditory and visual simple reaction times," *Neuroscience and Medicine*, vol. 01, no. 01, pp. 30–32, 2010.

[18] M. Bar, "The proactive brain: using analogies and associations to generate predictions," *Trends in Cognitive Sciences*, vol. 11, no. 7, pp. 280–289, Jul. 2007.

[19] P. Larsson, D. Västfjäll, P. Olsson, and M. Kleiner, "When what you hear is what you see: Presence and auditory-visual integration in virtual environments," in *Proceedings of the 10th annual international workshop on presence*. Citeseer, 2007, pp. 11–18.

[20] P. Vickers, "Ars informatica – ars electronica: Improving sonification aesthetics," in *Understanding and Designing for Aesthetic Experience: workshop at HCI 2005: The 19th British HCI Group Annual Conference*, 2005.

[21] W. W. Gaver, "What in the world do we hear?: An ecological approach to auditory event perception," *Ecological Psychology*, vol. 5, no. 1, pp. 1–29, mar 1993.

[22] P. Bergman, A. Sköld, D. Västfjäll, and N. Fransson, "Perceptual and emotional categorization of sound," *The Journal of the Acoustical Society of America*, vol. 126, no. 6, pp. 3156–3167, 2009-12.

[23] M. L. Eriksson, R. Atienza, and L. Pareto, "The Sound Bubble: A context-sensitive space in the space," *Organised Sound*, vol. 22, no. 01, pp. 130–139, mar 2017.

[24] F. Avanzini, M. Rath, D. Rocchesso, and L. Ottaviani, "Low-level models: resonators, interactions, surface textures," *The Sounding Object*, pp. 137–172, 2003.

[25] X. Pan and A. F. de C. Hamilton, "Why and how to use virtual reality to study human social interaction: The challenges of exploring a new research landscape," *British Journal of Psychology*, vol. 109, no. 3, pp. 395–417, Mar. 2018.

[26] J.-C. Servotte, M. Goosse, S. H. Campbell, N. Dardenne, B. Pilote, I. L. Simoneau, M. Guillaume, I. Bragard, and A. Ghuysen, "Virtual reality experience: Immersion, sense of presence, and cybersickness," *Clinical Simulation in Nursing*, vol. 38, pp. 35–43, Jan. 2020.

[27] K. Falkenberg, M. L. Eriksson, E. Frid, T. Otterbring, and S.-O. Daunfeldt, "Auditory notification of customer actions in a virtual retail environment: Sound design, awareness and attention," in *Proceedings of ICAD 2021*, 2021, in press.

[28] P. D. Thompson, J. G. Colebatch, P. Brown, J. C. Rothwell, B. L. Day, J. A. Obeso, and C. D. Marsden, "Voluntary stimulus-sensitive jerks and jumps mimicking myoclonus or pathological startle syndromes," *Movement Disorders*, vol. 7, no. 3, pp. 257–262, 1992.

PERIPHERAL AUDITORY DISPLAY FOR 3D-PRINTING PROCESS MONITORING

Maxime PORET (maxime.poret@labri.fr)[1], **Sébastien IBARBOURE** (s.ibarboure@estia.fr)[2], **Catherine SEMAL** (catherine.semal@ensc.fr)[3], **Myriam DESAINTE-CATHERINE** (myriam@labri.fr)[1], and **Nadine COUTURE** (n.couture@estia.fr)[2]

[1]*Bordeaux INP, CNRS, LaBRI, UMR5800*, **University of Bordeaux**, F-33400 Talence, France
[2]*ESTIA Institute of Technology, LaBRI, UMR5800*, **University of Bordeaux**, F-64210 Bidart, France
[3]*Bordeaux INP, CNRS, INCIA, UMR5287*, **University of Bordeaux**, F-33076 Bordeaux, France

ABSTRACT

When monitoring an industrial process, extreme sensory conditions can make it difficult to rely solely on direct observation. In this paper, we describe the development of an alternative display method for the production criteria of a wire-arc 3D-printing process using sonification. We made this display mostly ambient, as it is preferable in order to avoid fatigue in long-term usage. The sounds were chosen to be cognitively distinct progressive alarms so they would be easier to identify. The evaluation consists in a dual-task identification trial, so as to measure the proper communication of critical information as well as account for the level of distraction from other tasks. The results show that the attentional pull is rather minor and still allows for above-random criteria recognition rates. Though, there seems to be an occasional cognitive overlap between the sounds representing local and global overheating. The droning tone for the height of the part also tends to be drowned out in some cases. Both flaws will need to be addressed in future iterations.

1. INTRODUCTION

Despite considerable progress in the automation of industrial processes, a human presence still tends to be required to monitor the machines. This monitoring task can usually be carried out via simple visual observation. However, in practice, visual attention is not always guaranteed as operators may be distracted or focused on other more active tasks. Additionally, an industrial working context is likely to be too unfriendly on the senses to allow for direct observation.

Hearing tends to be more versatile and better adapted to perceiving changes over time than vision, while not requiring constant focus [1]. This makes it a suitable modality for real-time process monitoring by users faced with visually overwhelming working conditions [2–5], in order to avoid the pitfall of inattentional blindness [6, 7].

Such auditory displays of data can be achieved through sonification, a data-driven, non-verbal sound [8], usually produced through algorithmic processes in a "systematic, objective and reproducible" way [9]. The use of sonification for monitoring has been a subject of research for many years, in domains as varied as surgical gestures [10, 11], vital signs [12], business processes [13–15], internet activity [16–19], algorithmic processes [20], or domestic activity [21, 22].

While developing our sonification for a manufacturing process, we want to avoid the "better safe than sorry" approach of using sudden and loud alarms, as pointed out by Patterson et al. [23] and Lazarus et al. [24]. Instead we need a continuous sound that can be relegated to the background of other activities and evolve into a notification when necessary. This type of notification system is known as a peripheral display, or an ambient information system [25].

Our goal in this paper is to construct and evaluate a peripheral sonification prototype for the monitoring of an industrial 3D-printing process. As this work is still in an early stage of development, the evaluation will be conducted in a simulated work context rather than in-situ. We start by describing the process to be sonified as well as its use context. We then analyse the existing methodology regarding the design and evaluation of peripheral displays, before describing our prototype and its dual-task evaluation process. From the results, we assess ways to improve the sounds used.

2. 3D-PRINTING PROCESS

The process to be monitored is a wire-arc 3D-printing process [26]. Operators for those machines need to be able to detect anomalies in five criteria: the local width, height, and temperature monitored at the position of the printing head, and the global height and temperature along the part being constructed. See also [15].

The printing takes place inside an inert atmosphere to prevent chemical reactions that may impair the material's properties. Unfortunately, this precaution gets in the way of the operator's visual inspection. The wire-arc process emits flashing lights and projections, so operators have to wear protective masks which also greatly narrow down their fields of vision. Thus it is only really convenient to visually check the production during the cooling phases

between each layer. Even then, the discrepancies to be noticed in the geometry are usually smaller than a few millimeters, and the temperature cannot be assessed visually most of the time.

For these reasons, there has been an effort in the last few years towards augmenting reality for manufacturing processes using alternative display methods [15, 27–30] such as, in the case of this work, sonification for wire-arc 3D-printing.

We notice that, to some extent, the sounds of manufacturing already provide some insight into defects that may be occurring during the printing process, such as the noise grains becoming more distinct in case of a lower weld pool, or the sound stopping entirely in case of a material shortage. However, that sound is overall loud and unpleasant, as well as potentially dangerous for hearing upon prolonged exposure. Thus operators wear noise-reducing headphones to protect their ears. Our aim is to put those headphones to good use by having them output an auditory display designed to help monitor the process.

3. RELATED WORKS

3.1 Peripheral Displays

In 1985, Jenkins saw the potential in the hearing modality for information communication in ambient contexts [1]. The concept of ambient or peripheral displays then rose in popularity in the late 1990s and early 2000s with the arrival of ubiquitous computing and calm technologies announced by Weiser & Brown in 1996 [31]. In 1998, Wisneski et al. offered an early review on the topic, while calling for more research into ambient information technologies [32].

Such research took place in the 2000s in an effort to boil down the main criteria for the design of a peripheral display based on its goals and use context. McCrickard et al. [33] define 3 criteria: interruption, reaction and comprehension. Matthews et al.'s criteria [34] relate more to the way a notification should appear in one's field of attention: abstraction, notification level and transition. Pousman and Stasko [25] give 4 criteria: information capacity, notification level, representational fidelity and aesthetic emphasis. A few nuances aside, all these criteria can be roughly aggregated into the following list of considerations, which we used to better define the scope of our display:

- Information capacity: How many dimensions of data does the display need to account for? Here we have 5 dimensions (the weld pool's width, height, temperature, and the part's height and temperature). For all of those dimensions except the part's temperature, users should also be able to recognize the direction of the anomaly.

- Information abstraction: How precisely should users be able to reconstruct the data from the display? Here, there is no need for exact values but users need to know which dimensions are behaving abnormally, in which directions, and whether those anomalies should be considered critical.

- Notification levels: How does the degree of urgency evolve according to the type of information being conveyed? Here we want a subtle progression of the sounds following data fluctuations, so that a slight change in a dimension, without necessarily being detrimental to the production in itself, can preemptively catch the user's attention for the potential arrival of a bigger shift.

- Aesthetic emphasis: How pleasant should the display be? So far, the criteria for our work seem to relate it to what Pousman et al. call an "information monitor display", for which aesthetics are of rather low priority [25]. Though, since users would be listening to that sound repeatedly and over prolonged periods of time, we still feel it is necessary to make it pleasant enough to not become stressful.

3.2 Evaluation Methodology

A few different approaches can be taken to evaluate a peripheral display. Eventually, the best way is to put the display to use directly in its intended context by means of an in-situ implementation [22, 35]. Although, in early design stages, this is not always possible or suitable, either from a lack of equipment or because the display is still too experimental to be representative of what the intended audience may expect.

In a lot of situations, simply asking users to assess their experience through interviews and surveys is enough to gather information about the aesthetic value and intrusiveness of a display [36–40]. This is sufficient when the display's intended use is to be part of a relaxing augmented environment for the house, workplace, or public spaces.

Additionally, in cases where the display needs to convey more critical information, the evaluation also has to account for the intelligibility of that information. This requires more quantifiable data on users' performance when using the display, which are usually obtained by means of identification trials [15, 41, 42].

When a critical information display is intended to be part of a larger work context, a measurement of distraction is also needed. McCrickard et al. recommend a dual-task evaluation process to this end [33]. This methodology has also been researched more recently by Hausen et al. [43], Daniel [44], and it was implemented in several experiments on peripheral auditory displays [19, 21, 45–49].

In the case of our work, in-situ implementation is not feasible yet, as no sensors are actually present on the printers to provide the critical data to be monitored. Still, our goal is to produce a display that will help monitor the process with no need for direct exposure. This requires us to take into account other activities that would be made possible by this newfound sensory freedom, such as for example "checking one's e-mail" or "preparing the next print". Thus, our experiment will not only account for data intelligibility, but also for attentional capture through the use of a dual-task identification trial.

4. MAPPING CHOICES

Soundscapes of several simultaneous sound streams have been shown to facilitate the identification of multidimensional data [12, 47, 49–51] so we chose to convey our data using a soundscape of four perceptually and cognitively distinct sounds streams. The natural world offers many audible phenomena that can be metaphorically related to temperature (boiling, sizzling, exploding, crackling), but not that many when it comes to hearing the dimensions of an object. So, although we can afford to symbolically represent temperature with temperature-related auditory icons [52], the display of geometry requires a more abstract representation. For our display, we chose musical parameters. We expect that using sounds of such different natures will help quickly identify which one is behaving abnormally. Following is a description of how each sound stream is constructed and mapped to its corresponding criterion.

The geometric criteria (part height, weld pool width and weld pool height) are conveyed by continuous streams of structured, repetitive musical notes. It is preferable that those notes follow western rules of musical intervals, as they are easier to identify for european listeners [53], and are commonly considered more pleasant to listen to than atonal or noisy sounds. In the absence of anomalies, those notes constitute a baseline sound confirming that the sonification is up and running. As anomalies arise though, their fluctuations should induce a feeling of slight unease in the listeners, thus prompting reaction [54].

For the local weld pool dimensions, a lead arpeggio (L) of 3 notes in the chord of C major keeps playing as long as the dimensions are within bounds. This repetitive sequence of notes serves as a metaphor for droplets of matter being deposited during printing. The timbre for this sound is the default SuperCollider synth: a basic piano-like sound. The width influences the duration of those notes (inverse polarity mapping between 0.5 and 1.5 seconds). The height is conveyed by the starting pitch of the sequence (between C5 and F6). Loudness is also influenced by an amplitude factor, computed as the mean of two values respectively mapped to width and height anomalies (each between 0.02 and 0.2). We expect this sound to stand out in case of an anomaly by becoming faster, louder, and more erratic as the dimensions diverge from the norm.

For the relative part height (difference between the expected height and the current height), a continuous droning synthetic tone (D) varies in pitch (notes between E2 and D3). The absolute value of the height difference is conveyed by an amplitude factor mapped between 0.1 and 0.4. This continuous sound serves as a metaphor for the continuity of horizontal layers, with pitch fluctuations representing irregularities in a layer. The timbre for this sound is constructed as a sawtooth wave, bandpass-filtered around its first and second harmonics with each filtered harmonic playing in the left and right ear respectively.

Meanwhile, the thermal criteria (weld pool temperature and part temperature) are conveyed by noisy pre-recorded natural sounds that emerge in case of anomalies but remain silent otherwise. We elected to use the sounds of water reacting to heat and cold as they constitute an easily identifiable everyday metaphor for temperature in the system, and their noisy nature makes them stand out against the tonal background.

The weld pool temperature, when below its ideal value, is conveyed by the sound of crackling ice (W-). A temperature over the ideal value is conveyed by the sound of boiling water (W+). Straying further from the accepted range influences an gain factor mapped between 0 and 0.9, then rescaled and graduated as:

$$Gain = \begin{cases} 0 & \text{if } 0 \leq Gain \leq 0.4 \\ 0.2 & \text{if } 0.4 < Gain \leq 0.6 \\ 0.5 & \text{if } 0.6 < Gain \leq 0.9 \end{cases} \quad (1)$$

Thus this sound stream is inaudible as long as the temperature is within bounds, and only emerges as it turns into an anomaly.

Finally, when the global temperature of the part passes its threshold of 600°C, the sudden sound of sizzling water (S) is triggered.

The pitch, speed and loudness ranges for those sounds were chosen as a consequence of our previous work on the same project [15], which resulted in the participants requesting lower, slower and overall more distant sounds.

In the following sections, anomalies will be referred to by the first letter of their sound elements. For instance, the combination of lead arpeggio, drone, and boiling water anomalies will be called LDW+.

5. EXPERIMENT

5.1 Process

The primary task of our dual-task evaluation is based on the one described in [44]. It consists in copying random sequences of 'X' and 'O' symbols, whose lengths are randomly picked between 2 and 5. Participants interact with this game by clicking elements of a graphical user interface. As soon as a sequence has been copied, another one is generated and displayed, prompting the participants to copy as many sequences as they can in the duration of each level. We chose this very simple primary task because it gives an easily quantifiable assessment of the participants' performance while not relying too heavily on any one's individual abilities.

Simultaneously, the secondary task consists in listening and labelling sounds in real time by checking the corresponding boxes in the interface. See Figure 1. Those boxes are labelled after the types of sounds conveying the anomalies: "Lead", "Drone", "Water" and "Sizzle". W+ and W- are fused into a single box in the interface, simply labelled "Water" as, for now, the evaluation is more focused on the recognition of the anomalies than their polarities.

Before getting to the evaluation itself, players go through a progressive training phase during which they learn to copy sequences, then to recognize sounds, and finally to carry out both tasks at the same time. This training can be redone as many times as the player deems necessary. Still, players have to get a labelling score of 90% or higher

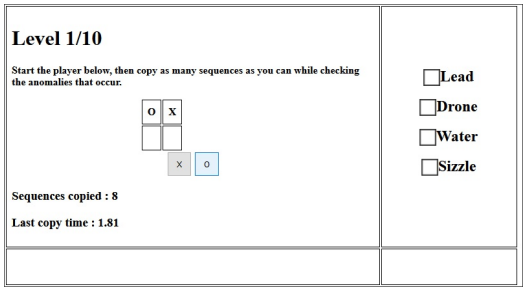


Figure 1. A screen capture of the experiment interface during a level. In the middle, the player rewrites the sequence displayed by clicking the 'X' and 'O' buttons in the same order. Upon each sequence completion, a new one appears. Boxes on the right allow the player to point out anomalies as he or she notices them.

in the last phase of that training before they can start the evaluation.

This evaluation interface can still be accessed online [1], but it does not record entries anymore.

5.2 Data

We used pre-simulated data recorded in .csv files representing various printing scenarios. Our data were sonified into .wav files according to the mapping choices described in Section 4 using a SuperCollider [2] script. In those simulations, the only anomaly combinations encountered are the ones that are likely to occur according to the way criteria physically interact (e.g. a higher local temperature causes the weld pool to spread out more, thus becoming lower and wider). This gives us 8 possible combinations, including the regular anomaly-free behaviour. Three of those were selected for the training phase and presented in this order: LD, LDW+, and no anomaly. All other anomaly combinations available were used for the experiment in a randomized order: LDW-, D, LW+, LW-, LW+S, and five more situations with no anomaly.

5.3 Participants

43 participants took part in the experiment: 20 M, 23 F, aged from 18 to 67 (average 32). By taking part in the experiment, participants certified that their hearing was unaltered. Five of them had taken part in an earlier experiment for the same project and were familiar with some of the mapping choices.

6. RESULTS

We measured participants' performance at the primary task by recording the length and time of completion of each sequence copied. For the secondary task, we recorded the times at which anomaly boxes were checked. After the experiment, participants were given the option to also answer a short survey on the aesthetics and intrusiveness of the display.

[1] https://maxime-poret.emi.u-bordeaux.fr/these/eval2020/ - Accessed 3/12/21

[2] https://supercollider.github.io/ - Accessed 3/12/21

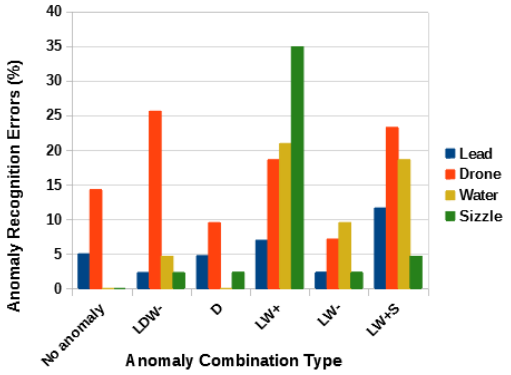


Figure 2. Mean error rate for the identification of anomalies, for each anomaly type (colors) and for each anomaly combination (horizontal sections).

6.1 Anomaly identification

For each type of level, we computed the error rate for anomaly identification, such that a criterion was considered inaccurately identified when its box was checked despite there being no anomaly, or unchecked despite the presence of an anomaly. Those results are displayed in Figure 2. We find it encouraging that all criteria were recognized above random chance, as it is likely that with more training testers would be able to identify all anomalies more accurately. Still, the most frequent errors highlight which parts of the display can be made clearer in future iterations.

D seems to be the most difficult anomaly to label as its error rate is the highest in 4 levels out of 6. For levels LW+, LW-, LW+S, and no anomaly, false positives may be due to the fact that people start expecting D for every anomaly combination, as it is often linked to others and is present in most of the training levels. In levels LDW- and D, false negatives may be due to the fact that the drone is more subtle than the other sounds, and can be more easily tuned out or drowned out. Both false positives and false negatives seem to indicate that the drone sound is not noticeable enough for some testers, who instead choose to respond seemingly "at random".

We also notice that, in the level LW+, the sound of boiling water was sometimes mistaken for the sizzle, which resulted in 35% of testers checking that box. During LW+S, the sizzle was mostly recognized but some participants neglected the L and W+ anomalies also occurring at the same time.

6.2 Attentional curves

We computed the attentional curves for each type of level as the average symbol-copying speed of participants over the course of a level. On the same time scale, we also plotted the anomaly onsets and average labelling times as

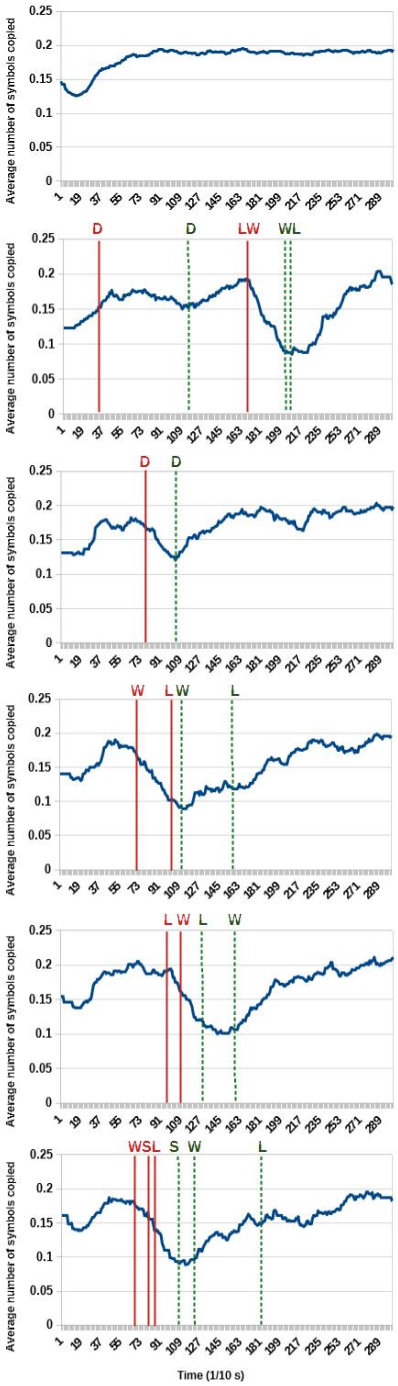


Figure 3. Attentional curves for each level type in the evaluation, computed as the average number of symbols copied for each decisecond. Red lines: onset times of the anomalies. Green dotted lines: mean annotation time. Level types from top to bottom: No Anomaly, LDW-, D, LW+, LW-, LW+S

timestamps of the attentional capture of the participants. See Figure 3.

In levels with no anomaly, users get gradually more efficient at the primary task as their copying speed reaches a limit of 0.2 symbols per decisecond after 8 seconds. A similar dynamic can be observed at the start of the other types of levels, but with an efficiency drop of approximately 0.1 symbols per decisecond when an anomaly is triggered. Participants do not seem to have issues recovering once they have reacted, since by the end of each level the average copying speed returns to the limit of 0.2 observed in levels with no anomaly. Recovery appears to take more or less time depending on the number of onsets, their distribution in time and their durations.

In the LDW- level, the anomaly onset for D did not affect participants' performance as much as the anomalies in most levels (about 0.03 symbols per decisecond instead of 0.1). Although it was still noticed on average before L and W- started playing, it took longer to be labelled than most of the anomalies. This may be due to the fact that, in that level, the drone's pitch starts slowly lowering before any other anomaly is triggered, which may be more difficult to perceive than faster changes, or a rising pitch.

In the LW+S level, although W+, S and L were triggered in this order with delays of 1 second between each, S was the first one to be attended to on average, possibly due to its more startling nature and its relative rareness in the experiment.

6.3 Survey

After testing the display, 21 of the participants also answered a survey about their experience. In the survey, they were presented with a series of sentences regarding the experiment, which they could rate on a scale from 1 (disagree) to 3 (agree), 2 being a neutral response. 18 participants (85.7%) disagreed with the sentence "The sound bothered me while doing the task", while the rest remained neutral. On the sentence "I found the sound to be stressful", 14 participants (66.7%) disagreed, 6 (28.6%) remained neutral, and 1 (4.8%) agreed. These answers suggest that the sound was not perceived as overly intrusive by testers, but that its aesthetics, especially when it comes to inducing stress, could be more polished. A more formal evaluation of these rather qualitative properties of the display is still to be produced.

7. CONCLUSION

We produced an auditory display for an industrial process that does not allow for direct visual monitoring. This display is intended to be minimally-intrusive and aesthetically pleasing. The sound streams were chosen in a way that should make them easily identifiable and relatable to the criteria they represent. We evaluated this display with a focus on both the attentional pull and the intelligibility of the information.

Our experiment shows that there is an overlap between the sounds of sizzle and boiling water that makes it more difficult for users to distinguish them when they are pre-

sented separately. In expected use scenarios, though, the sizzle sound is mostly intended as a last resort alert. Indeed, it should not occur very often and the sound of boiling should have already been playing for a good amount of time when the sizzle happens. We find it encouraging that, although both sounds were not perfectly discriminated, most testers definitely recognized overheating alerts.

We also find that when the drone's pitch goes downward too slowly, it is harder to notice as an anomaly, so a linear mapping of relative height to pitch alone may not be the most suitable choice. We could make this sound stream more alerting by having another timbre emerge when the part height passes its tolerated threshold.

The brief evaluation process we implemented gives us insight into flaws that can be addressed in future iterations of the prototype, but it would also be interesting to know how many of the reoccurring mistakes would still be made after a longer training period, possibly over several sessions.

Predictably, most anomaly onsets cause the attention for the primary task to drop, but participants are still able to recover rather quickly. It is worth noting that not everyone takes the same amount of time to move their mouse between the two areas of the screen. This adds a bias to our computation of attention which we could have measured in an early step of the experiment (for instance by timing testers clicking back and forth between those areas) and accounted for in the results.

Sound ecology is an important aspect of auditory monitoring [55] that we wish could have been more thoroughly taken into account in both the design and evaluation of the display. Indeed, despite the use of noise-reducing headphones, it is unlikely that the noise of production will be entirely suppressed, which may get in the way of some of the sounds we chose. Also, due to the sanitary conditions at the time of testing, the evaluation was presented as a webpage sent out to participants, who all played it at home on their own setups and using their own sound gear. For those reasons, we look forward to experimenting in better standardized conditions in the future.

Once improved for optimal recognition rates, this display is intended to be put to use in further experimentation on integrating sonification into an augmented work context, putting operators in simulated printing sessions where the criteria are displayed through both sound and touch.

Acknowledgments

The authors of this paper would like to thank the SCRIME of the University of Bordeaux for hosting this research, as well as Matthias Robine and Emmanuel Duc for their involvement in the previous iteration of this work. We would also like to thank the Addimadour platform and Anaïs Domergue for allowing us to observe a wire-arc 3D-printer in action in February of 2020 and answering our questions.

8. REFERENCES

[1] J. J. Jenkins, "Acoustic information for objects, places, and events," in *Persistence and Change: Proceedings of the First International Conference on Event Perception*, W. Warren, Ed. Hilldale, NJ, USA: Lawrence Erlbaum, 1985, pp. 115–138.

[2] J. W. Marcum, "Beyond visual culture: the challenge of visual ecology," *portal: Libraries and the Academy*, vol. 2, no. 2, pp. 189–206, 2002.

[3] S. Oviatt, "Human-centered design meets cognitive load theory: Designing interfaces that help people think," in *Proceedings of the 14th ACM International Conference on Multimedia (MM06)*, Santa Barbara, CA, USA, 2006, pp. 871–880.

[4] Q. Wang, S. Yang, M. Liu, Z. Cao, and Q. Ma, "An eye-tracking study of website complexity from cognitive load perspective," *Decision Support Systems*, vol. 62, pp. 1 – 10, 2014.

[5] J. Zagermann, U. Pfeil, and H. Reiterer, "Measuring cognitive load using eye tracking technology in visual computing," in *Proceedings of the Sixth Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization (BELIV '16)*, Baltimore, MD, USA, 2016, pp. 78–85.

[6] R. Marois, D.-J. Yi, and M. M. Chun, "The neural fate of consciously perceived and missed events in the attentional blink," *Neuron*, vol. 41, no. 3, pp. 465–472, 2004.

[7] J. M. Wolfe, T. S. Horowitz, and N. M. Kenner, "Rare items often missed in visual searches," *Nature*, vol. 435, no. 7041, pp. 439–440, 2005.

[8] G. Kramer, B. N. Walker, T. Bonebright, P. Cook, J. H. Flowers, and N. Miner, "The sonification report: Status of the field and research agenda. report prepared for the national science foundation by members of the international community for auditory display," in *International Community for Auditory Display (ICAD)*, Santa Fe, NM, 1999.

[9] T. Hermann, "Taxonomy and definitions for sonification and auditory display," in *Proceedings of the 14th International Conference on Auditory Display (ICAD2008)*, Paris, France, 2008.

[10] T. Ziemer, D. Black, and H. Schultheis, "Psychoacoustic sonification design for navigation in surgical interventions," *Proceedings of Meetings on Acoustics*, vol. 30, no. 1, p. 050005, 2017. [Online]. Available: https://asa.scitation.org/doi/abs/10.1121/2.0000557

[11] S. Matinfar, T. Hermann, M. Seibold, P. Fürnstahl, M. Farshad, and N. Nassir, "Sonification for process monitoring in highly sensitive surgical tasks," in *Proceedings of the Nordic Sound and Music Computing Conference 2019 (Nordic SMC 2019)*. Stockholm, Sweden: KTH: In Press, 2019.

[12] W. T. Fitch and G. Kramer, "Sonifying the body electric: Superiority of an auditory over a visual display

in a complex, multivariate system," in *Santa Fe Institute, Studies In the Sciences Of Complexity Proceedings*, G. Kramer, Ed., vol. 18. Reading, MA, USA: Addison-Wesley Publishing Co., 1994, pp. 307–326.

[13] T. Hildebrandt, "Towards enhancing business process monitoring with sonification," in *Business Process Management Workshops (BPM 2013)*, N. Lohmann, M. Song, and P. Wohed, Eds. Beijing, China: Springer International Publishing, 2014, pp. 529–536.

[14] T. Hildebrandt and S. Rinderle-Ma, "Toward a sonification concept for business process monitoring," in *19th International Conference on Auditory Display (ICAD 2013)*. Lodz, Poland: Lodz University of Technology Press, 2013, p. 323–330.

[15] M. Poret, S. Ibarboure, M. Robine, E. Duc, N. Couture, M. Desainte-Catherine, and C. Semal, "Sonification for 3d printing process monitoring," in *Proceedings of the 2021 International Computer Music Conference (ICMC 2021)*, Santiago, Chile, 2021, forthcoming.

[16] M. Gilfix and A. L. Couch, "Peep (the network auralizer): Monitoring your network with sound," in *Proceedings of the 14th USENIX Conference on System Administration (LISA'00)*. Berkeley, CA, USA: USENIX Association, 2000, pp. 109–118.

[17] M. H. Hansen and B. Rubin, "Babble online: Applying statistics and design to sonify the internet," in *Proceedings of the 7th International Conference on Auditory Display (ICAD2001)*. Helsinki University of Technology, Espoo, Finland: International Community on Auditory Display, 2001, pp. 10–15.

[18] C. Chafe and R. Leistikow, "Levels of temporal resolution in sonification of network performance," in *Proceedings of the 7th International Conference on Auditory Display (ICAD2001)*. Helsinki University of Technology, Espoo, Finland: International Community on Auditory Display, 2001, pp. 50–55.

[19] M. Barra, T. Cillo, A. De Santis, U. F. Petrillo, A. Negro, and V. Scarano, "Multimodal monitoring of web servers," *IEEE MultiMedia*, vol. 9, no. 3, pp. 32–41, 2002.

[20] E. Childs, "The sonification of numerical fluid flow simulations," in *Proceedings of the 7th International Conference on Auditory Display (ICAD 2001)*, Helsinki University of Technology, Espoo, Finland, 2001.

[21] Q. T. Tran and E. D. Mynatt, "Music monitor: Ambient musical data for the home," in *Proceedings of the IFIP WG 9.3 International Conference on Home Oriented Informatic and Telematics (HOIT 2000)*, A. Sloan and F. van Rijn, Eds., vol. 173. Kluwer, 2000, pp. 85–92.

[22] G. M. Vallejo Rosas, "Listenin: Ambient auditory awareness at remote places," Ph.D. dissertation, Massachusetts Institute of Technology, 2003.

[23] R. D. Patterson, T. F. Mayfield, D. E. Broadbent, A. D. Baddeley, and J. Reason, "Auditory warning sounds in the work environment," *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, vol. 327, no. 1241, pp. 485–492, 1990.

[24] H. Lazarus and H. Höge, "Industrial safety: Acoustic signals for danger situations in factories," *Applied Ergonomics*, vol. 17, no. 1, pp. 41–46, 1986.

[25] Z. Pousman and J. Stasko, "A taxonomy of ambient information systems: Four patterns of design," in *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI06)*. Association for Computing Machinery, 2006, pp. 67–74.

[26] S. W. Williams, F. Martina, A. C. Addison, J. Ding, G. Pardal, and P. Colegrove, "Wire + arc additive manufacturing," *Materials Science and Technology*, vol. 32, no. 7, pp. 641–647, 2016. [Online]. Available: https://doi.org/10.1179/1743284715Y.0000000073

[27] A. Olwal, C. Lindfors, J. Gustafsson, T. Kjellberg, and L. Mattson, "Astor: An autostereoscopic optical see-through augmented reality system," in *Proceedings of International Symposium on Mixed and Augmented Reality*, 2005.

[28] A. Ceruti, A. Liverani, and T. Bombardi, "Augmented vision and interactive monitoring in 3d printing process," in *International Journal on Interactive Design and Manufacturing*, 2017.

[29] J. Wang, J. Erkoyuncu, and R. Roy, "A conceptual design for smell based augmented reality: Case study in maintenance diagnosis," *Procedia CIRP*, vol. 78, pp. 109–114, 2018, 6th CIRP Global Web Conference – Envisaging the future manufacturing, design, technologies and systems in innovation era (CIRPe 2018). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827118312472

[30] A. Hattab and G. Taubin, "Rough carving of 3d models with spatial augmented reality," in *Proceedings of the ACM Symposium on Computational Fabrication*. ACM, Jun. 2019. [Online]. Available: https://doi.org/10.1145/3328939.3328998

[31] M. Weiser and J. S. Brown, "The coming age of calm technology," 1996.

[32] C. Wisneski, H. Ishii, A. Dahley, M. Gorbet, S. Brave, B. Ullmer, and P. Yarin, "Ambient displays: Turning architectural space into an interface between people and digital information," in *International Workshop on Cooperative Buildings (CoBuild'98)*. Darmstadt, Germany: Springer, 1998, pp. 22–32.

[33] D. S. McCrickard, C. M. Chewar, J. P. Somervell, and A. Ndiwalana, "A model for notification systems evaluation—assessing user goals for multitasking activity," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 10, no. 4, pp. 312–338, 2003.

[34] T. Matthews, A. K. Dey, J. Mankoff, S. Carter, and T. Rattenbury, "A toolkit for managing user attention in peripheral displays," in *Proceedings of the 17th annual ACM Symposium on User Interface Software and Technology (UIST04)*, Santa Fe, NM, USA, 2004, pp. 247–256.

[35] N. Sawhney and C. Schmandt, "Nomadic radio: Speech and audio interaction for contextual messaging in nomadic environments," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 7, no. 3, pp. 353–383, 2000.

[36] S. E. Hudson and I. Smith, "Electronic mail previews using non-speech audio," in *Conference Companion on Human Factors in Computing Systems (CHI96)*, Vancouver, British Columbia, Canada, 1996, pp. 237–238.

[37] E. R. Pedersen and T. Sokoler, "Aroma: Abstract representation of presence supporting mutual awareness," in *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI97)*, ser. CHI '97. Atlanta, GA, USA: Association for Computing Machinery, 1997, pp. 51–58.

[38] E. D. Mynatt, M. Back, R. Want, M. Baer, and J. B. Ellis, "Designing audio aura," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI98)*, Los Angeles, CA, USA, 1998, pp. 566–573.

[39] F. Kilander and P. Lönnqvist, "A whisper in the woods - an ambient soundscape for peripheral awareness of remote processes," in *Proceedings of the 8th International Conference on Auditory Display (ICAD 2002)*, Kyoto, Japan, july 2002.

[40] N. Rönnberg, J. Lundberg, and J. Löwgren, "Sonifying the periphery: Supporting the formation of gestalt in air traffic control," in *Proceedings of The 5th Interactive Sonification Workshop (ISon-2016)*, CITEC, Bielefeld University, Germany, 2016, pp. 23–27.

[41] T. L. Bonebright and J. H. Flowers, "Evaluation of auditory display," in *The Sonification Handbook*, T. Hermann, A. Hunt, and J. G. Neuhoff, Eds. Logos Berlin, 2011, pp. 111–144.

[42] B. S. Mauney and B. N. Walker, "Creating functional and livable soundscapes for peripheral monitoring of dynamic data," in *Proceedings of the 10th International Conference on Auditory Display (ICAD 2004)*, Sydney, Australia, 2004.

[43] D. Hausen, A. Tabard, A. Thermann, K. Holzner, and A. Butz, "Evaluating peripheral interaction," Munich Germany, 02 2014, pp. 21–28.

[44] M. Daniel, "Shape-changing Cylindrical Displays: Application to Data Physicalization and Peripheral Interaction for Energy Demand-Side Management," Theses, Univ. Bordeaux, Nov. 2018. [Online]. Available: https://hal.archives-ouvertes.fr/tel-01972386

[45] R. Jung, "Ambience for auditory displays: Embedded musical instruments as peripheral audio cues," in *Proceedings of the International Conference on Auditory Display (ICAD 2008)*. Paris, France: International Community for Auditory Display, 2008.

[46] C. Ho and C. Spence, "Using peripersonal warning signals to orient a driver's gaze," *Human Factors*, vol. 51, no. 4, pp. 539–556, 2009, pMID: 19899363. [Online]. Available: https://doi.org/10.1177/0018720809341735

[47] T. Hildebrandt, T. Hermann, and S. Rinderle-Ma, "A sonification system for process monitoring as secondary task," in *2014 5th IEEE Conference on Cognitive Infocommunications (CogInfoCom 2014)*. Vietri sul Mare, Italy: IEEE, 2014, pp. 191–196.

[48] T. Hermann, T. Hildebrandt, P. Langeslag, and S. Rinderle-Ma, "Optimizing aesthetics and precision in sonification for peripheral process monitoring," in *Proceedings of the 21st International Conference for Auditory Display (ICAD 2015)*, Graz, Austria, 2015, pp. 317–318.

[49] T. Hildebrandt, T. Hermann, and S. Rinderle-Ma, "Continuous sonification enhances adequacy of interactions in peripheral process monitoring," *International Journal of Human-Computer Studies*, vol. 95, pp. 54–65, 2016.

[50] S. Smith, R. M. Pickett, and M. G. Williams, "Environments for exploring auditory representations of multidimensional data," in *Santa Fe Institute, Studies In the Sciences of Complexity Proceedings*, G. Kramer, Ed., vol. 18. Reading, MA, USA: Addison-Wesley Publishing Co., 1994, pp. 167–184.

[51] L. Demany, M. Erviti, and C. Semal, "Auditory attention is divisible: Segregated tone streams can be tracked simultaneously," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 41, no. 2, p. 356, 2015.

[52] W. W. Gaver, "What in the world do we hear? an ecological approach to auditory event perception," *Ecological Psychology*, vol. 5, no. 1, pp. 1–29, 1993.

[53] A. J. Watkins and M. C. Dyson, "On the perceptual organisation of tone sequences and melodies," *Musical Structure and Cognition*, pp. 71–119, 1985.

[54] P. N. Juslin and P. Laukka, "Expression, perception, and induction of musical emotions: A review and a questionnaire study of everyday listening," *Journal of New Music Research*, vol. 33, no. 3, pp. 217–238, 2004.

[55] P. Vickers, "Sonification for process monitoring," in *The Sonification Handbook*, T. Hermann, A. Hunt, and J. G. Neuhoff, Eds. Logos Berlin, 2011, pp. 455–492.

CARETUNES: TURNING PATIENT VITALS INTO MUSIC

Elif ÖZCAN (e.ozcan@tudelft.nl)(0000-0001-9195-2960) [1], **Chen CHOU**[1], **Koen BOGERS**[1], and **Aadjan van der HELM**[1]

[1]*Critical Alarms Lab, Faculty of Industrial Design Engineering*, **Delft University of Technology**, the Netherlands

ABSTRACT

This paper presents CareTunes as a concept to explore musical sonification of patient vitals and the role of music in Intensive Care Units (ICU). In this paper, we first describe the design specifications for the sonification of data in a musical fashion. Secondly, we present two applications for CareTunes in prototype stage and user evaluation studies. The first application regards the ICU nurses' need to monitor patients from a distance (CareTunes as musical updates for nurses) and second application regards families' need to connect with their loved in the ICU (CareTunes as musical messages for families). We conclude that music has the potential to represent changes in patient vitals for nurses and emotionally regulate families' anxieties regarding ICU patient's condition. Music offers a platform for reutilizing patient data for human-centered solutions.

1. INTRODUCTION

CareTunes is a concept that challenges the clinical utilization of data from patient monitoring devices found in Intensive Care Units (ICU). **CareTunes** is developed to explore methods for continuous musical stream that summarizes patient vital signs and presents them in a coherent, logical, and pleasant way to the clinicians (mainly the nurse) and families. In this paper, we first describe the technical requirements for CareTunes and its design specifications for the sonification of data. Secondly, we present two applications for CareTunes in prototype stage and user evaluation studies. The first application regards the ICU nurses' need to monitor patients from a distance (CareTunes as musical updates for nurses) and second application regards families' need to connect with their loved in the ICU (CareTunes as musical messages for families). Overall, this paper demonstrates our vision and pose critique on sound design in healthcare by discussing the role of music in ICUs and novel ways of using healthcare data.

Our motivation to explore music also comes from the need to eliminate cacophony and introduce harmony in the soundscapes of complex work spaces and healing environments. Essentially, we are exploring ways of sonification and its manifestation in the form of music which can offer richer data representations and pave the road for customization of sonified data by providing a systemic approach for data representation with its own rules. Music is also a familiar concept among people as daily musical interactions allow potential users to gain a perceptual repertoire and quickly interpret notation or infer abstract meaning.

1.1 Intensive Care Units

People are admitted to the Intensive Care Unit (ICU) when they are in a life-threatening situation, and special equipment is used to constantly monitor, support, and/or take over their bodily functions. ICU patients often require much rest, and some may be sedated, such as patients on mechanical ventilators. The average ICU length of stay is around 3.3 days [1]. However, it is influenced by several factors, such as the type of disease or surgery. For example, the average length of stay of COVID-19 patients is estimated to be around seven to eight days at the time the research was conducted [2]. The COVID-19 situation calls attention to people who are hospitalised for a longer period in the ICU. Families of ICU patients often experience distress and anxiety. They have emotional and social needs such as assurance and closeness, but oftentimes they cannot make contact with the patient, due to the medical restrictions or because the patient is unconscious. Such difficulties are emphasized during the COVID-19 pandemic, as patients may be transported to hospitals far from home due to the limited capacity of ICUs. This points to a demand for a stronger patient-family connection and connectedness. Nurses working in the ICU are exposed to a vast amount of sounds from medical equipment. The amount of alarms nurses cope with causes alarms fatigue, which causes nurses to become desensitised to alarms. Not only is this a threat to patient safety, it also causes stress.

1.2 The need for music in the ICU

Music can serve as a design approach for healthcare [3] as it is a powerful way to communicate emotions and meanings and it also exerts physical and behavioral effects [4]. With the power to express meaning, it is possible to use music to pass on information about the patient to the clinicians or family. Music can also be used to enhance or reduce positive or negative emotions, or simply to regulate levels of arousal [5]. Therefore, music has the quality to fulfill the social needs, emotional needs, and need for information at the same time. Furthermore, some events may have a more natural representation in sound [6]. Bly stated that: "Because perception of sound is different than visual perception, sound can offer a different intuitive view of the information it presents." [7] While numbers, texts, and the image of a loved one being unconscious may feel unfamiliar to their families, music may allow people to connect with the patient in a more natural way.

1.3 Communicating Information with Music

Research into music communication and Sonic Interaction Design (SID) offers guidance for designing music to communicate information [6] [8] [9]. For example, sound is better suited to display relative data than absolute data with high precision [6]. Therefore, if music is used to display everchanging information, such as the physiological data of the patient, it is not necessary, nor aesthetically pleasing, for the music to change precisely with every detail of the data it translates. Musical communication is not only about the music providing information, but also about people interpreting the music. Factors that may affect musical communication include musical features, situations and contexts, and individual preferences and knowledge [10]. Therefore, the meaning of the music may not be fixed, but rather changeable as fluids within certain boundaries, in relation to the listener's experience [11]. The mental space for imagination and interpretation can have its advantages and disadvantages. For example, it can make the listening experience more intimate as it allows the listener to give meanings to the music based on perception of the context in an emotional way. Nevertheless, the listener's individual preferences or emotions can greatly influence their interpretation of the music's meaning, making it difficult to design accurately for an intended effect using music. We conclude that the strength of music communication lies in its ability to facilitate the listener to perceive information in a humanized way with emotions considered.

1.4 Influencing emotions through music

Justin and Västfjäll [12] concluded six mechanisms through which music listening may induce emotions: *i.* brain stem reflexes (e.g., arousal feelings when hearing sudden, loud, dissonant sounds), *ii.* evaluative conditioning (memory of music stimulus paired with other positive or negative stimuli), *iii.* emotional contagion, *iv.* visual imagery, *v.* episodic memory (relating music with a past event), and *vi.* musical expectancy (e.g., surprise caused by changes). Instead, the transitions in music are designed to be gradual. Examples in using music to cause emotional contagion include the use of different timbres. For instance, the timbre of string instruments has a higher emotional quality due to its voice-like characteristic [5]. While the strings are able to evoke more longing, sadness and tenderness, piano can evoke more joy [13]. Furthermore, music may reduce anxiety by making people attuned to its melodies and harmonies. Lee et al. [14] concluded that the characteristics of anxiety-reducing music include *simple repetitive rhythms, predictable dynamics, low pitch, slow tempos, the consonance of harmony, a lack of percussive instrumentals and vocal timbres*. However, there has been no consistent agreement on the type of music, and people's musical preferences remain one of the most important factors in the effect of the use of music as a design tool [14].

1.5 Timbre and tempo as design principles

Patient data are continuous by nature and fluctuate upon metabolismic events. In a sonification of patient data, key, tempo, pitch and timbre could all be used to represent values of the different parameters involved. It is the sudden or drastic changes in the values that are monitored and acted upon. An important criterion for the design is whether or not listeners will likely be able to detect changes in sonification by foreground and background effects. Timbre seems to be an aspect of sound that is quite universally recognised by people. People are able to remember the voice of someone they know and recognise it. Differences between musical instruments, especially from different families, can be heard by most untrained people. A guitar clearly sounds different than a trumpet, even when playing the same note. In 'Remembering the melody and timbre, forgetting the key and tempo', Schellenbach and Habaschi [15] show how people perform better at the memorisation of melodies when the timbre remains the same in both instances they heard it in. This implies timbre plays a role in recognition of melodies. Wolpert [16] even shows that to people without musical training, timbre has a bigger influence on the perception of a musical excerpt than harmonics; when non-musicians were asked to select the excerpt they heard earlier, 95% selected a sample in which the instrumentation is unchanged, ignoring that sample's incorrect harmonic accompaniment. If non-musicians are indeed inclined to recognise musical samples by their timbre, this would be a useful attribute for to represent data in a sonification. It would mean users could listen to the sonification and recognise changes in the values sonification represents without having to identify tonal or harmonic changes. This can make the design less reliable on musical training.

Levitin and Cook [17] show that the memory of a piece of music contains a reasonably accurate tempo for that piece; when people were asked to reproduce a song they had not heard for at least 72 hours, the majority of people were able to reproduce the tempo of the song with an error of only 8% or less. This suggests there is an absolute memory of tempo when people remember a song. A musical sonification will inherently have a certain tempo. Because people have a reasonably accurate memory of a tempo of a musical piece they have heard before, this tempo, like timbre, seems a suitable aspect of music for the representation of data.

2. PROTOTYPING CARETUNES

2.1 Sonifying patient vitals

A first prototype is built to demonstrate the concept of using three of a patient's vital signs to influence the way different parts of music sound. We examined 48 hours of anonymized patient data from Drager Patient Monitors to characterize patient data flow. The data offer changes in patient vitals, which then input different software synthesisers. The musical prototype that is presented as a result the first cycle consists of a Max MSP program (Figure 1) that sonifies three patient vitals that are often monitored on the patient monitor: heart rate (HR), Oxygen levels in the blood (SpO2), and blood pressure (BP). Manipulating synthesisers based on data can be done with the help of Max MSP. Max also offers integration with Ableton Live, which will become instrumental in prototyping later design

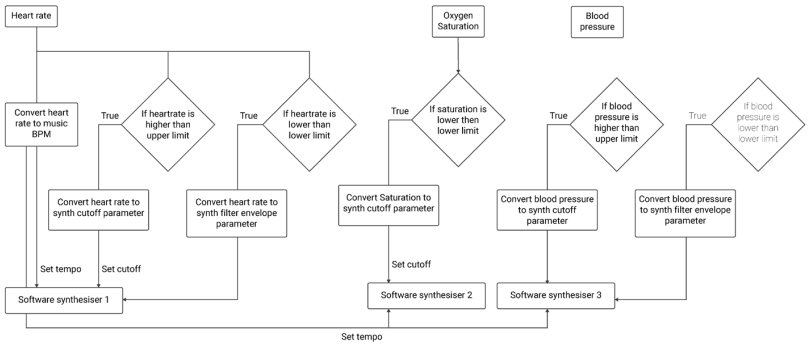


Figure 1. The state diagram for the Max patch.

iterations. In this first design attempt, software synthesisers and audio effects that are integrated in Max are used to generate musical motives. Parameters of the software synthesisers that influence its timbre can be manipulated based on the numbers of simulated patient vitals. Figure 1 shows a state diagram of how the Max patch works.

2.2 Musical complexities.

A second prototype was built to demonstrate the concept of switching between different complexities in the musical composition and also to explore the role of individual musical taste. Therefore, Ableton Live is selected as the most suitable tool for this prototype. Ableton Live is a program for music sequencing and arranging (Ableton.com). Ableton Live is preferred mostly because of its integration with Max, which makes it possible to automate parameters in audio effects from Max. In the 'session view' in Live, one can record audio or midi 'tracks' that all contain 'clips. These clips can be triggered, either in the interface itself or from external hardware, allowing the user to create a composition out of pre-recorded samples or midi sequences. Three simple compositions, changing in genre (i.e., pop, jazz, ambient), are written in Ableton Live all consisting of a chord progression accompanied by drums and a bass line. Within a composition**,** there are four different versions of each composition, each of which is slightly more complex than the previous one. This is achieved by adding extra bass notes, adding accents in the drum part or by playing the notes of the chord sequentially and more often. Musical scores for the two out of four intensities of one of the 'genres' are shown in Figure 2.

2.2.1 Evaluation of the concept.

Our aim was to understand whether the participants would be able to recognise the changes in timbre and whether they would need to change between songs and musical complexities for a long-term listening (effect of tempo). Genres were used to address individual music preferences.

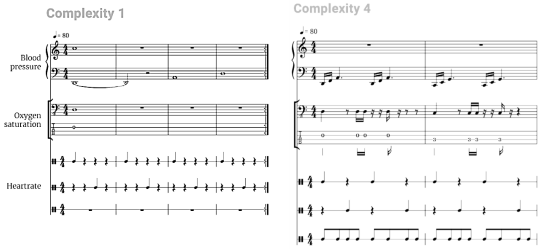


Figure 2. Broken chord and more elaborate drum and bass rhythms make the piece sound more complex.

Participants. Seven design students (4 males and 3 females) participated in the evaluation with a mean age of 23,5 years.

Set-up and procedure. The prototype that simulates the current version of the CareTunes concept played from Ableton Live on a laptop. The participant would listen to the continuous song (i.e., musical sonification) from this laptop while working. The participant was able to control the prototype through the Open Sound Control interface (OCS). The researcher triggered changes in timbre from the computer, which the participant could then react by pressing a button on OSC. Participants were given the task to recognise changes in timbre and indicate them on the mobile device. If they would prefer a more exciting version of the song they chose which would increase its 'complexity' on the OSC. Participants were also asked to indicate when they want to listen to a different sonification on the mobile device via OSC, and a new song started playing once they do so. Participants were occupied with another task while they listened to the songs for the duration of the test (i.e., one hour) to see whether the changes in the music could be detected when people are distracted.

A semi-structured interview was conducted after the test to get a grasp of how well participants are able to recognise different timbres and how confident they feel in doing so. The following questions were used in this interview. What is the participant's overall impression of the sonification? How hard was it for the participant to recognise and distinguish the changes? How quickly did the sonification start to bore the participant? Were the sonifications distracting from the work participants were trying to do?

Measurements. Timestamped notes of the user's behaviour are recorded including when they change the intensity or when they seem in doubt about how to respond to the soundscape, the number of times the user successfully and unsuccessfully recognises a change in timbre. The users verbally answer the interview questions.

Results. The results from the user test are divided into the measured results and the participants' answers to interview questions. The amount of time that participants were able to listen to the songs before they wanted to switch to a next one is reflected by the measured results of their behaviour. Participant quotes from the interview show how they felt about the different songs and complexities and switching between them. Figure 3 shows the participant's behaviour in switching between songs and what they thought about this. A large majority of timbre changes was successfully recognised by participants (Table 1). Participant quotes show how they experienced the task.

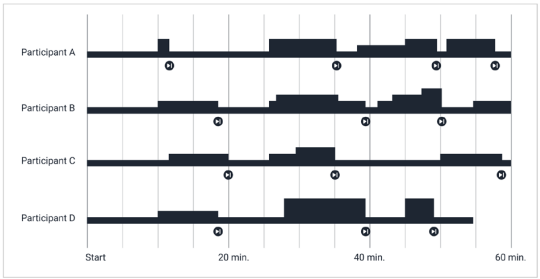


Figure 3. The moments participants changed songs (see the 'play' icon) during 60 minutes and their specific comments. The thickness of the lines changes as participants seek for a more complex musical structure.

Song-like sonifications can make it hard to consistently recognise changes. As one listens to a song, it is quite normal to hear changes in timbre as a part of it. Instruments may be played in different ways, or sound effects may be applied by the artist. The song-like structure in the current design therefore confuses people in whether the changes they hear are a part of the song or if they hear something they are supposed to point out.

Switching between songs can make it more interesting to listen to the soundscape for a long time, but can also make it harder to get used to using the sonification. Participants in the test were able to switch between different soundscapes and they did so roughly every ten minutes. The amount of time one can comfortably listen to the sonification can be increased by adding more songs. Some participants did however notice that they they had to get used to the new song to easily recognise timbre changes again.

Songs have parts that are layered, sounding at the same time, making it harder to distinguish them. Using a song like structure in the sonifications means each part representing a datapoint may sound at the same time as another part. Some of the participants were very easily able to detect a change in the song overall, but found it difficult to point out which of the part it was.

There is a learning curve to using CareTunes. Most participants note that it becomes easier to work with the sonification as they are listening to them for a while. This indicates working with a song may require a certain amount of training by the nurse.

	Hit/miss	*Participant quotes*	
Drums	*20/23*	*"One of the changes was a bot more subtje then the others. The drums were really clear."*	*"That change was not really disturbing. I thought it was just a different [effect] pedal or something."*
Bass	*22/23*	*"At first the changes were hard to recognise, but I got used to it and it became easier."*	*"Changing the intensity just makes me doubt if I heard a change or not."*
Piano	*16/25*	*"Doesn't sound very alarming"*	

Table 1. Participants' success rate in recognizing changes in timbre.

3. CARETUNES – MUSICAL UPDATES

Medical alarms are quite often misused as nurses look for possibilities to be constantly aware of the situation. The basic premise of an alarm use is to be notified when action needs to be taken [18]. Alarms seem to be often used by nurses as a constant status report. Nurses can do this by setting limits more tightly than they need to be, to be notified when the parameter approaches the actual limit. This use of alarms is the cause of many non-actionable alarms. Because of this, nurses can get used to switching off alarms right after they occur without attending to the patient. Currently, alarms used for monitoring patients do not carry the kind of information nurses may practically need [19]. Nurses need knowledge about several values measured by equipment to judge how they need to respond. Current alarms never give them this information except for the very worst cases, like an asystole alarm, is it immediately clear what action needs to be taken. All the less severe alarms (i.e., yellow alarms) may mean that any of the patient's vital signs slightly exceeds the value that has been set as a limit. It does not convey which vital sign has exceeded the limit, by how far it has done so, or even whether it surpassed the low or the high limit. This means nurses nearly always need to verify the state of the patient by checking the exact values, often to find out no action needs to be taken.

3.1 Design considerations for musical updates

The song-like structure of the sonification in the previous cycle made it hard for users to distinguish between the different parameters they were listening to. A brainstorm session with three trained musicians was organised to fulfill nurses' needs in relation to the sonification design. Several

requirements for the sonification were used as a starting point for the session: *i.* Each instrument used in the sonification to represent a different parameter should be easily distinguishable; *ii.* Four levels of timbre change on each side of the safe middle should be recognisable by the listener. These should have clearly audible differences between values higher and lower than the middle value; *iii*. The timbre of each parameter should become less pleasant to listen to as it approaches the limit set for that parameter; *iv*. Sounds which represent a value over the limit should universally be regarded unpleasant to listen to; *v*. Instead of continuous music, the sonification should give the listener periodic updates.

Timbre change. For each parameter, the range between the upper and lower limit is divided into four segments above the safe middle and four segments below it. Each of these segments has its own timbre so the listener can identify each segment by listening. This allows the nurse to be aware of how much change occurs in one direction or the other. As the parameter approaches a low limit, the sound that represents that parameter will sound more muffled in each segment. This is achieved through the use of a low pass filter over the original, safe value sound. Each of the segments the frequency of the filter is changed just enough to create a noticeable difference from the segment that precedes it. The same technique is used to make each segment approaching an upper limit sound slightly sharper. In this case a high pass filter is used to achieve this effect.

Dissonance. The timbre changes indicate a parameter that reaches an unsafe value and help the nurse be more proactive in taking action as a patient's vitals change. When a limit that a nurse has set is crossed by a parameter however, this should still be made very clear to the nurse, as a dangerous situation may be the result of this. To ensure the nurse's attention is gained in such a situation, dissonance is used in the sonification. Dissonance in musical theory are notes that sound unstable together and need resolution. Out of key notes are added to the sound that represent the parameter that exceeds the limit. Out of rhythm sounds are added when this concerns the heart rate parameter. To look for the right solutions for these requirements, the session involved ideations with the help of Ableton Live and several MIDI instruments. A range of instruments and sound effects were tested to find those that could be used to create the desired amount different timbres. The result of the session was an overview of how changes in each of three vital functions will influence an audio effect. The parameters that are sonified were divided not just across different instruments, but across three different aspects of music: rhythm, harmony and melody. The three parts play in sequence to make them easier to distinguish. The listener hears three medical parameters, one by one (See Figure 4).

3.2 Usability test with nurses

A usability test with nurses was conducted to get more in-depth insights in how well CareTunes works and how it is experienced by nurses. The questions addressed by the user test are: What values do nurses intuitively assign to the different timbres in each parameter they hear in Care Tunes? How well are nurses able to recognise the different parameters and the changes within them? Do the visual cues in the user interface support nurses in understanding what they hear? How do nurses feel about using CareTunes in their work? What do they see as advantages and disadvantages? Would they trust themselves to hear changes in the sonification?

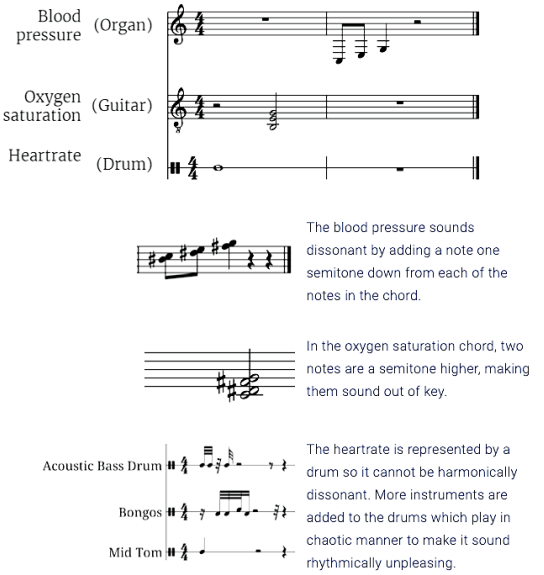


Figure 4. An out of limit value is indicated by dissonance.

Participants. Four ICU nurses (two male, two female) with the same roles but different levels of experiences participated.

Procedure. Before starting the test, the basic principles of the design were explained to nurses. A storyboard was used to help the participant understand the role that CareTunes would fulfill in the ICU. The test started by the participant listening to the sonification first without and then with the GUI. The participant listened through on-ear headphones which allowed for conversation to take place while listening. While the participants listened to the simulated patient data, they were asked explain what they heard and what it could mean. When all timbre changes have been covered, a participant was debriefed.

Results and Conclusions. Two boards are used to show the difference between listening without and with the GUI. Quotes are linked to the different timbres that the nurses heard during the session, showing how they intuitively perceive them, and how this changed when the GUI was introduced. This gives the following insights.

Some of the timbres in each parameter intuitively sound urgent to participants. For other parameters, some participants feel the timbre change may be too subtle. Most participants were able to recognise the timbre change corresponding with a value close to the limit. All participants recognised a value crossing the limit. The introduction of

visual support clearly made it easier for participants to follow the parameters and hear which part of the music belongs to each of them. When listening to the sonification with visual support, some participants indicate the sonification sounds less urgent than they would expect for the heartrate and oxygen saturation parameters. Three participants felt wearing an earpiece would be convenient way of listening to CareTunes and one expressed discomfort and concern about nurses wearing hearing aids.

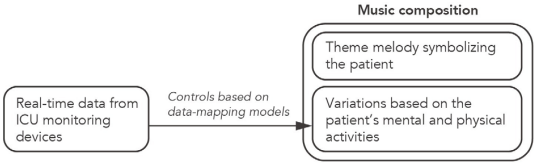
4. CARETUNES – MUSICAL MESSAGES

The music design of CareTunes as musical messages for Families focuses on emotional contagion and visual imagery. Emotional contagion as a concept can help us think of way to reduce negative emotions and evoke positive emotions, and visual imagery helps to communicate information regarding the patient state. Sudden changes in the music can avoided to prevent causing surprises or sudden emotional arousal. The data sources are selected according to what kind of information would better increase the sense of connectedness, and what the families would be likely to interpret or imagine (See Figure 5). This information includes the patient's emotions, state of consciousness, and activities. Therefore, heart rate, brain waves, and movements serve as data sources for the music variations. Heart rate and brain waves are data that are on a spectrum. For example, a normal resting heart rate for adults ranges from 60 to 100 beats per minute (though some ICU patients' heart rate can be higher due to increased metabolic rate [20]), and brainwaves are defined by electroencephalography (EEG) frequency: alpha wave (8-13Hz), theta wave (3-7Hz), etc. [21] Therefore, they are divided into levels when being presented in the music.

A mixture of meanings. Although the patient data sources are separate, they usually affect one another, often in a complex manner, and cannot be distinctly separated to represent respective meanings. For example, when a person starts exercising, their heart rate and movements would increase. Nevertheless, an increase in heart rate does not necessarily mean a person is exercising; it can also signify emotional arousal such as stress or excitement, as heart rate. Heart rate and brain waves also change according to a person falling asleep or waking up.

Limitations and alternatives. While heart rate is always monitored, continuous EEG monitoring is yet uncommon and still being developed in the context of ICU [22]. Motion sensors are not a standard equipment in the ICU, either. Nevertheless, this project included these data sources as a design vision for the ideal situation. If it is to be implemented without EEG and motion monitoring devices, alternatives can be considered, such as using heart rate to present the state of consciousness and physical activities. However, this may not have the same effect as using brain waves, because families might relate thoughts and emotions more to the brain, and regard the heart to have a more "biological" meaning, according to the user evaluation. It is also possible to eliminate the presentation of physical activities and focus on presenting psychological activities. Meanwhile, this also points to one of the advantages of having multiple tracks of music in the design: it allows different ICUs to customize the music according to their resources and expertise.

Design Principles. The data derived from the monitoring devices should not be constantly changing to avoid evoking the feeling of instability and hence uncertainty. Therefore, the data is updated at regular intervals (e.g., every five minutes). In addition, if any track is to be added or removed from the music, it needs to follow the bars of the music and wait for the right moment to come in.



Data-mapping examples

Data source	Representation	Variations
Heart rate	Emotional arousals, sleep and wakefulness, physical activities	3 levels of liveliness by adding different music tracks
Brain waves (EEG)	Mental activities and state of consciousness	4 levels of key change from low to high pitch

Figure 5. Data mapping of the music generation.

4.1 User evaluation with families

Two rounds of design and qualitative user evaluation provided insights into how music can better present the patient to the family while reducing negative emotions and inducing positive emotions.

Procedure. In the user tests, the participants answered a short questionnaire after listening to each piece of music, then participated in a structured interview based on their answers to the questionnaire. In the first round of user evaluation, the participants listened to three pieces of music; in the second round, the participants listened to four variations of one piece of music, and were able to access the music within 48 hours.

Participants. In order to gain in-depth feedback, a total of eight participants were selected to join the qualitative user evaluation. The first user test consisted of six participants (three males and three females, mean age 27,5 years), three with experience of a relative being in the ICU. The second user test consisted of two participants whose close relatives have stayed in the ICU.

Data Collection and Analysis. The data collection of the research is through structured interviews. In the interviews, the participants first explained their answers to a self-report questionnaire about their emotions affected by the music. The questionnaires mainly serve to guide the participants to reflect on their emotions and experience. The participants then proceeded to talk about how they interpreted the music, and what they thought the music told

them about their loved one in the ICU. The interview results are arranged into codebooks with different themes and sub-themes to present the commonality of emotional effect by music on people.

Results and Conclusions. Tables 2 and 3 show the results of the user evaluation interviews. In the first round of user evaluation, participants expressed that they feel more connected to their loved ones when the music feels more positive and tender. Furthermore, the continuous stream of music felt like the continuous presence of the patient. Participants also pointed out that, when the music directly presents the crucial information, such as the heartbeat of the patient, they feel worried and uncertain. Meanwhile, the music feels more meaningful when they present the patient's emotions and activities. In the second round of user evaluation, more patient data are included in the music composition. Participants expressed that music enables them to feel calmer and more positive in critical situations. They also feel that the music enhances intimacy between them and their loved ones as music presents the emotions of the patient. The family could also bring in their own imagination to interpret what the music represents.

Theme	Description	Examples
Evoking positive emotions	Pleasantness in music can evoke the feeling of connectedness.	P4: "I feel connected when the music is positive and clear."
	Tenderness in the music can help calm the family.	P3: "I like the music, because of the slow tempo, which is good for relaxing."
	Intimacy can be evoked from music with more details and variations.	P4: "The feeling of intimacy evoked by music B was stronger than A, because music A was calm and plain, while music B has more elements with buzzing and details."
Presenting the patient	Continuous stream of music enables people to feel the presence of their loved one.	P2: "The continuous stream of music is like a representative of them, enabling me to feel them."
	Using music similar to heartbeats to represent the patient can evoke anxiety.	P1: "I worried if the music was going to slow down or go faster. [...] Using tempo to present the heartbeat felt too literal."
	People may find it meaningful when the music is more related to emotions and activities.	P1: "It feels meaningful if it is more related to emotions and not as a way to monitor." P4: "The patient in music B might be conscious. They are active and moving around, and awake, while [the patient in] music A feels like sleeping."
Meaning of the music	People worry about misinterpreting the music when the music presents crucial information.	P2: "I want to know what the music and rhythm actually mean. [...] I want to know what to expect and what means 'bad'." P3: "I might misinterpret the music. [...] I am worried that if the heartbeat slows down, the music would change." P6: "When a new sound appears I wonder what it means."

Table 2. Results of the first round of user interviews.

5. CONCLUSIONS

In this study, we explored the role of music in the ICU and its potential for reutilizing patient data in a way that corresponds to the psychological and emotional needs of ICU users. We studied patient data and found ways to represent it musically. Our explorations provided evidence that music has the power to represent changes in patient vitals when used as an update tool for nurses replacing yellow non-actionable alarms and emotionally regulate families' anxieties when used a messaging tool. Our prototypes prioritized possibilities for sonification and did not tackle actual system design, which will be the priority for the next design iteration for CareTunes in the near future.

Furthermore, we did not study the effect of musical data representation on patients. Live music has shown to be effective in reduction of pain. Patients own musical data may have a positive effect on regulation heart rate and variations. Following studies will look into physiological effect of music on patients, what music could mean to different types of patients (neonates, children, and adults) and how individual music preferences can be dealt with. Music as a therapeutic tool will be so investigated. In addition, this is an explorative study in sonification and its clinical relevance must be tested in randomized controlled trials.

Theme	Description	Examples
Evoking positive emotions	The music is able to help the family stay calm.	P2: "It sounds calm. [...] He is not recovering, but i wouldn't feel very sad because the emotion can be controlled. [...] I feel assured because the chords are brighter. [...] I feel calm because the pace is slow and not anxious. [...] I could face it and accept it."
Presenting the patient	Families may picture the patients differently when listening to their music.	P1: "It feels like she is peacefully lying on a field of grass." P2: "It feels like he is recovering, and not so lonely and bored."
Meaning of the music	People become less worried about their interpretation when the music does not present crucial information.	P1: "The atmosphere is good, and I would not analyze the music carefully. I would not worry if anything bad happens."
Data for generating the music	Families find the music meaningful when it is generated from brain waves.	P1: "Anything the person generates is 'real'. Heart represents the sign of life, but brain waves bring in emotions. [...] The brain-waves-generated music makes me assured."
Music composition	The multiple music tracks with different characteristics help construct multidimensional scenes or stories.	P1: "I thought of the meadow because of the light and crisp sounds. [...] The foreground is lively with flowers and birds, and the background is steady. [...] There are also big waves which feel like the wind." P2: "The [bell] sounds are like external stimulations [...] and the background sounds describe his feelings."

Table 3. Results of the second round of user interviews.

Acknowledgments

We collaborated with Yoko Send of Sen Sound (USA) and Joseph Schlesinger, MD of Vanderbilt University MC (USA) with the Design United (NL) funding granted to Elif Özcan. We thank all the participants and professional musicians Ruben Nosaehi (NL) and Nicolas Laget (USA) who contributed to the studies and prototyping. The Care-Tunes projects were MSc theses of Koen Bogers (Care-Tunes for nurses https://www.youtube.com/watch?v=tgHE4niHZag) and Chen Chou (CareTunes for family; https://www.youtube.com/watch?v=t2dLM_Lbays) at the Critical Alarms Lab of TU Delft. For listening the pieces of CareTunes or click on the links above.

6. REFERENCES

[1] Hunter, A., Johnson, L., & Coustasse, A., "Reduction of intensive care unit length of stay: the case of early mobilization," in The health care manager, 2014, pp. 128-135.

[2] Health Information and Quality Authority (HIQA), "Evidence summary for average length of stay in the intensive care unit for COVID-19," 2020. https://www.hiqa.ie/reports-and-publications/health-technology-assessment/evidence-summary-average-length-stay [Accessed May 2, 2020].

[3] Özcan, E., Frankel, L., & Stewart, J., "Uncommon music making: The functional roles of music in design for healthcare," in Music and Medicine, 2019, pp. 245-255.

[4] Hargreaves, D. J., MacDonald, R., & Miell, D., "How do people communicate using music," in Musical communication, 2005, pp. 1-26.

[5] Sakka, L. S., & Juslin, P. N., "Emotion regulation with music in depressed and non-depressed individuals: Goals, strategies, and mechanisms," in Music & Science, 2018.

[6] Franinović, K., & Serafin, S. (Eds.), Sonic interaction design. Mit Press, 2013.

[7] Bly, S., Sound and computer information presentation. Lawrence Livermore National Lab., CA (USA); California Univ., Davis (USA), 1982.

[8] Rocchesso, D., Serafin, S., Behrendt, F., Bernardini, N., Bresin, R., Eckel, G., ... & Visell, Y., "Sonic interaction design: sound, information and experience," in CHI'08 Extended Abstracts on Human Factors in Computing Systems, 2008, pp. 3969-3972.

[9] Özcan, E., van Egmond, R., Gentner, A., & Favart, C., "The Case of Toyota Motor Europe," in Foundations in Sound Design for Embedded Media: A Multidisciplinary Approach, 2019.

[10] Inskip, C., MacFarlane, A., & Rafferty, P., "Meaning, communication, music: towards a revised communication model," in Journal of Documentation, 2008.

[11] Feld, S., "Communication, music, and speech about music," in Yearbook for Traditional Music, 1984, pp. 1-18.

[12] Juslin, P. N., & Västfjäll, D., "Emotional responses to music: The need to consider underlying mechanisms," in Behavioral and brain sciences, 2008, pp. 559-575.

[13] Lahdelma, I., & Eerola, T., "Single chords convey distinct emotional qualities to both naïve and expert listeners," in Psychology of Music, 2016, pp. 37-54.

[14] Lee, O. K. A., Chung, Y. F. L., Chan, M. F., & Chan, W. M., "Music and its effect on the physiological responses and anxiety levels of patients receiving mechanical ventilation: a pilot study," in Journal of clinical nursing, 2005, pp. 609-620.

[15] Schellenberg, E. G., & Habashi, P., "Remembering the melody and timbre, forgetting the key and tempo," in Memory & cognition, 2015, pp. 1021-1031.

[16] Wolpert, R. S., "Recognition of melody, harmonic accompaniment, and instrumentation: Musicians vs. nonmusicians," in Music Perception, 1990, pp. 95-105.

[17] Levitin, D. J., & Cook, P. R., "Memory for musical tempo: Additional evidence that auditory memory is absolute," in Perception & Psychophysics, 1996, pp. 927-935.

[18] Sanz-Segura, R., & Özcan, E., "Alarm Response in Critical Care: Obstacles for Compliance," in International Conference on Healthcare Ergonomics and Patient Safety, 2019, pp. 73-81.

[19] Edworthy, J. R., McNeer, R. R., Bennett, C. L., Dudaryk, R., McDougall, S. J., Schlesinger, J. J., ... & Osborn, D., "Getting better hospital alarm sounds into a global standard," in ergonomics in design, 2018, pp. 4-13.

[20] Kara, D., Akinci, S. B., Babaoglu, G., & Aypar, U., "Increased heart rate on first day in intensive care unit is associated with increased mortality," in Pakistan journal of medical sciences, 2016, p. 1402.

[21] National Sleep Foundation, "Stages of Human Sleep. The Sleep Disorders," 2020. https://sleepdisorders.sleepfoundation.org/chapter-1-normal-sleep/stages-of-human-sleep/ [Accessed August 10, 2020]

[22] Caricato, A., Melchionda, I., & Antonelli, M. (2018). "Continuous electroencephalography monitoring in adults in the intensive care unit," in Critical Care, 2018, p.75.

A HISTORICAL ANALYSIS OF HARMONIC PROGRESSIONS USING CHORD EMBEDDINGS

Elia ANZUONI (elia.anzuoni@epfl.ch) [1*], **Sinan AYHAN** (sinan.ayhan@epfl.ch) [1*],
Federico DUTTO (federico.dutto@epfl.ch) [1*],
Andrew MCLEOD (andrew.mcleod@epfl.ch) (0000-0003-2700-2076) [1],
Fabian C. MOSS (fabian.moss@epfl.ch) (0000-0001-9377-2066) [1], and
Martin ROHRMEIER (martin.rohrmeier@epfl.ch) [1]

[1] *Digital and Cognitive Musicology Lab*, *Digital Humanities Institute*, **École Polytechnique Fédérale de Lausanne**, Switzerland
[*] Authors 1, 2, and 3 contributed equally to this work.

ABSTRACT

This study focuses on the exploration of the possibilities arising from the application of an NLP word-embedding method (Word2Vec) to a large corpus of musical chord sequences, spanning multiple musical periods. First, we analyse the clustering of the embedded vectors produced by Word2Vec in order to probe its ability to learn common musical patterns. We then implement an LSTM-based neural network which takes these vectors as input with the goal of predicting a chord given its surrounding context in a chord sequence. We use the variability in prediction accuracy to quantify the stylistic differences among various composers in order to detect idiomatic uses of some chords by some composers. The historical breadth of the corpus used allows us to draw some conclusions about broader patterns of changing chord usage across musical periods from Renaissance to Modernity.

1. INTRODUCTION

Algorithmic approaches to music usually come in two flavors: music information retrieval (MIR) aims at extracting relevant patterns from musical signals (e.g. audio recordings, MIDI files, or images of scores) and improve the performance on certain specific tasks, such as genre or composer classification, automatic playlist generation, optical music recognition and more. Computational music analysis, on the other hand, aims at using data-driven methods to study the domain of music in order to develop a deeper understanding of its cultural and historical diversity, or implications for its perception and cognition.

This study bridges the two approaches by applying the machine-learning (ML) methods often employed for the task of chord prediction in MIR to a large corpus of symbolic chord sequences. However, our goal is not to globally optimize chord prediction in this dataset. Rather, we use the chord-prediction task as a benchmark measure for investigating stylistic characteristics of different composers in the dataset. We suppose that the historical dimension in particular affects stylistic differences which, in turn, should be reflected in the performance of a (globally constant) chord predictor. In other words, assuming a fixed model for chord prediction, how does its performance change given historically varying input? What conclusions can we draw from this perspective?

In the remainder of the paper, we first summarize recent related work (Section 2). We then describe the dataset used in our study (Section 3), as well as our specific application of the three ML approaches in more detail (Section 4). We report two important results (Section 5): that clustering in an embedding space reveals functional relations between chords, and that changes in performance of our chord-prediction model (dependent on composer and historical time) indicate fundamental changes in the usage of harmony.

2. RELATED WORK

Our study draws on a dataset of symbolic musical chord sequences and uses three fundamental machine learning building blocks: word embeddings, clustering, and Recurrent Neural Networks (RNNs).

Word embedding is a popular technique in Natural Language Processing (NLP) which learns a mapping of words to vectors in a low-dimensional *embedding space* from a *corpus* of texts, which is supposed to contain sufficient information on the semantic relationships between words. The mapping is such that the relative positions of the vectors (hopefully) reflect these semantic relationships. The precise learning of this mapping is dependent on the specific method used. We use Word2Vec [1]. In Word2Vec, words often appearing in similar contexts are mapped to close points in the embedding space, according to their cosine distance.

Previous work has used Word2Vec successfully for modeling aspects of the musical language. In [2], the authors show that a simple approach of splitting musical scores into short slices containing note presence information is able to capture some simple features such as tonal proximity. Later, in [3], a similar slicing procedure is used on a

larger corpus that re-affirms Word2Vec's ability to model musical concepts such as tonal relationships between musical keys. In [4], the authors learn an embedding space in a similar way, but use as input multi-hot vectors of note presence, rather than one-hot encodings of unique symbols (as in the standard Word2Vec). In contrast to these efforts, our work takes annotated chord symbols as input, thus enabling us to model information at a much higher level of abstraction by eliminating information spurious to the harmonic structure such as short passing tones and ornamentation.

Clustering is a well-known unsupervised learning primitive, which works by grouping together close points in a space, and is used to extract information about the points that might be contained in their coordinates. We use hierarchical clustering [5] with cosine distance to analyze the structural properties of our resulting chord embedding space. This hierarchical approach (as opposed to a more naive clustering approach like K-means [6]) has the benefit of allowing us to investigate clusters at different levels of granularity without needing to fine-tune any hyperparameters. Previous work has also investigated the clustering of musical embeddings, using explicitly trained chordal embeddings (e.g., [2,3]), chord clusters induced through training for a different task (e.g., [7, 8]), or clustering of larger groups of chords (e.g., [9]).

RNNs are widespread tools in NLP, particularly in the field of word prediction with their Long Short-Term Memory (LSTM) [10] variant. LSTMs are particularly suited to this task because of their structure, involving a *forget gate*, which solves the *short-term memory* problem, typical of traditional RNNs. Similar work shows how they can be successfully employed in musical contexts, for "next-slice" modeling [4, 11], as well as for chord prediction [7, 12], and cadence identification [13]. While the cited works try to maximize prediction accuracy as much as possible, our goal is slightly different. Of course, we do want the models to perform as well as possible, but our main focus in the current work is instead to investigate the change in prediction performance across historical time (enabled by our expansive corpus), and to try draw musicological conclusions from this.

3. DATA

The dataset at our disposal, used for embedding, clustering, and chord prediction, consists of 4045 chord progressions in pieces by 24 Western classical composers, spanning the wide historical range from the Renaissance to 20th-century Modernism. The data has been derived from harmonic annotations using the syntax presented in [14–17]. For this study, the labels have been simplified in order to decrease the size of the chord vocabulary and to remove sparsity in our data. The pieces have been partitioned into local key segments that are either in the major or the minor *mode* (i.e., they contain no modulations), and chords are expressed relative to the tonic of that mode. Specifically, chords are represented by their *root* (expressed as a Roman numeral referring to the scale degree of the mode) and their *quality* (major, minor, diminished, or augmented; 7th chords are reduced to their corresponding triad). Because of this representation, the chord vocabulary is *potentially* infinite because the seven scale degrees of the two modes can be preceded by arbitrarily many accidentals. In particular, this allows us to distinguish enharmonically equivalent triads, such as `#II:MAJ` and `bIII:MAJ` that may entail different harmonic functions. Applied chords have been reduced to be directly related to the tonic of the mode, e.g. "vii°/V" is translated to "♯iv°" and represented as `#IV:DIM`. Thus, the chord sequences in our dataset are of the form

- `MAJOR;I:MAJ,II:MIN,V:MAJ,...`, or
- `MINOR;I:MIN,II:DIM,III:MAJ,...`,

where mode and chord labels are separated by a semicolon and chords within a progression are separated by commas. The average length of a chord sequence is 31 chords for major sequences and 28 chords for minor sequences. Since the roots of chords are expressed in relative notation, i.e. as the distance to the tonic, an F major chord is represented as `IV:MAJ` if the chord sequence is in C major, but as `III:MAJ` if it is in D minor. Following these reductions, there are 81 distinct chords in major sequences, and 77 different chords in minor sequences in our data.

As one can observe in Figure 1, the amount of data at our disposal varies greatly across composers and historical periods. Note, for example, that no chord sequences in the major mode are available for Sweelinck. Great care has thus to be taken when generalizing our results to the entire œuvre of these composers or the historical periods they represent. The data is available at `https://github.com/DCMLab/chordembeddings-smc2021`.

4. METHODOLOGY

4.1 Chord embedding

Our first processing step, serving as a basis for the two downstream tasks of clustering and chord prediction, is the application of Word2Vec [1]—specifically its implementation in the Gensim library [18]—which takes as input "sentences" (in our case, major or minor sequences) of "words" (in our case, chord labels). We treat major and minor chord sequences as independent and never include chord sequences from both modes in conjunction. Thus, in the following, when we say "train/test on all sentences/sections of a composer" or "train/test on a composer", we implicitly mean that those sections are all in the same mode.

Word2Vec has four hyperparameters to tune: `size`, `window`, `sg` (skip-gram), and `min_count`. `size` determines the dimension of the embedding space. To avoid overfitting, it should be less than the size of the vocabulary, i.e. the number of distinct chords in the corpus. In our case, the vocabulary size varies considerably, between 20 and 100 chord types per composer within either of the two modes. `window` defines the "width" of the context, i.e. how many chords, to the left and to the right, constitute the context of the current chord. The binary parameter `sg`

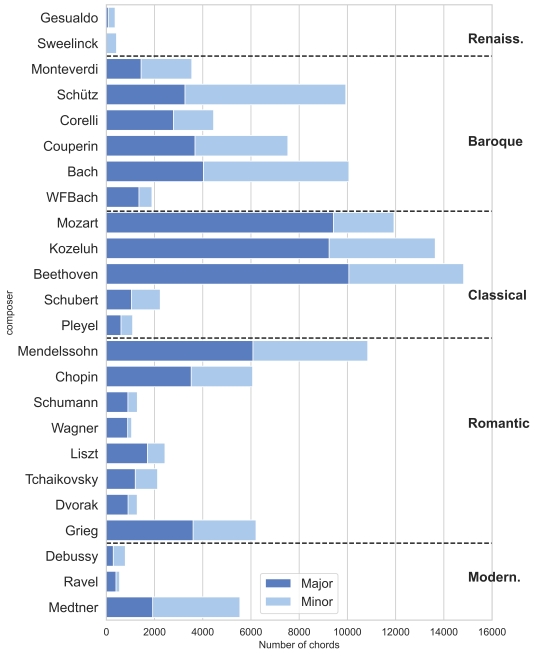


Figure 1. Total number of non-unique chord labels used by each composer, split between major and minor sequences. Composers are ordered by year of death (from oldest at the top to more recent at the bottom).

is short for "skip-gram" and selects the training algorithm: it can be either "continuous bag of words" (CBOW, i.e. guessing the target word from its context), or "skip-gram" (guessing the context given the target word). `min_count` sets a minimum absolute frequency a chord must have in order to be kept in the corpus. Since our corpus contains a Zipf-like distribution, this allows us to remove from the result the numerous irrelevant mappings of rare chords.

For all of our experiments, we exclude rare chords, as the model is unable to learn a stable embedding for such chords, making any relevant conclusion impossible. We therefore set `min_count` $= 50$ (since the most common chords have absolute frequencies of hundreds, if not thousands), which led to a vocabulary size of 32. The `size` of the embedding space was then chosen to be 5 (alternatives were essentially equivalent). We set `window` $= 2$ (again, other values led to similar results), and finally, we chose to use skip-gram rather than CBOW embeddings, because this led to more interpretable results.

4.2 Clustering

A first application of the mapping learned by Word2Vec is clustering, which is used to detect musical patterns. As is understandable from the properties of the mapping, chords appearing in the same cluster are likely to often appear in similar contexts. For this task, it is very difficult to carry out an objective, quantitative model evaluation. Therefore, we choose hyperparameters based on how much the outcome corresponds to music-theoretical intuitions. For example, we expect, when only training on major sections, that tonics and dominants are embedded close to each other, since they constitute the most basic musical pattern imaginable, as discussed in [15], and therefore often occur in very similar contexts.

Hierarchical clustering works by recursively merging the pair of clusters C_i and C_j (starting from singletons) that are the closest to each other according to some distance metric. We use *cosine distance*, commonly used for vector embedding spaces. The recursion stops when the minimum distance between clusters is above a given `distance_threshold`, or when only a single cluster remains.

The fact that this algorithm can work with cosine distance is ideal to detect similarities in a Word2Vec embedding space. Moreover, it is able to capture clusters of any shape. One might argue that a choice of `distance_threshold` can be quite arbitrary. However, this can be avoided by setting the `distance_threshold` to some large value (thus merging all clusters into one), and then plotting a dendrogram of all possible mergers. A dendrogram (e.g., Figure 3) is a depiction of the nested clusters produced by this method: it clearly shows all the mergers $C_i - C_j$ that happened, and the distance associated to them.

4.3 Chord prediction

Another use of the mapping provided by Word2Vec is the chord prediction task. LSTMs are an improvement over the classic RNN design that solve its *short-term memory* problem (caused by the well-known *vanishing gradient* problem): this allows them to effectively track long-term dependencies in sequential data. They are commonly used in NLP to predict the next word in a sentence.

We implemented an LSTM-based neural network for chord prediction, which trains on a *training corpus* (all sentences from a set of *training composers* for a given mode) and is tested on a *test corpus* (all sentences from a single *test composer* for that mode). For the LSTM experiments, the Word2Vec embedding is retrained using only the training corpus. Thus, we test how well-predictable chords in musical sequences by a composer are given the knowledge about chord sequences by all other composers. The metric used is the simple accuracy: the fraction of correctly-predicted chord occurrences, either overall or grouped by chord. We use the overall accuracy results for a single test composer to see how "predictable" they are, from what we learned from the training composers. We use the same results, split by chord, to investigate which chords are easier to predict and which are used more idiomatically (and are thus more difficult to predict).

The LSTM design is shown in Figure 2, and is structured as follows. Given a target chord (c_n in the figure), a first LSTM layer takes as input the concatenation of the embedded vectors of chords within some window of the target chord (shown as black circles in the figure with a window size of 2). A linear layer then maps the LSTM's output vector to a vector of length `n_vocab` (where `n_vocab` is the number of distinct chords), with a final softmax activa-

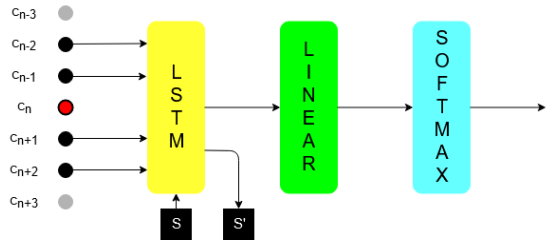


Figure 2. Diagram for the predictor network.

tion.

For the chord prediction experiments, we use the same Word2Vec parameters as above, although the embeddings are recalculated for each test composer. For both training and testing the LSTM, we take care to remove any data points which contain any chord (either as input or as the target) that falls below Word2Vec's `min_count` (in the training corpus). For training the LSTM, we use the Adam optimizer [19] with mean squared error (MSE) for the loss. We train all results for 2 epochs (this was enough for them to converge in all cases).

5. RESULTS

Our results imply two main findings: 1) clustering chords in the embedding space reveals meaningful functional relations between many of them; 2) chord prediction accuracy exhibits historical trends.

5.1 Clustering reveals functional chord relations

First, we report the results we obtained by applying hierarchical clustering on the embedded chords from the major and minor sections of all composers in the corpus. We visualize the hierarchical clustering in the embedding spaces for the major and the minor mode in dendrograms in Figures 3 and 4, respectively. As mentioned before, distances in embedding spaces are inherently difficult to interpret in general. However, many of the resulting clusters are quite well interpretable in various ways.

The resulting clusters for chord sequences for both modes reveal two fundamental tonal relations: functional *equivalence* and functional *difference* [20–22]. This extends earlier similar findings on functional categories restricted to J. S. Bach's chorales and based on chord bigrams [23]. Below we list a number of notable functional chord relations that can be found in our clusterings.

5.1.1 Functional equivalence

Chords that share common tones may be regarded as functionally equivalent. Functionally equivalent chords include *relative* and *parallel* chords, as well as other *common-tone* relations [24]. Two chords are each other's relative if they are the tonics of two keys that have the same key signature (e.g. `V:MAJ` and `III:MIN` in a major key). A major and minor chord are parallel if they have the same root (e.g. `II:MAJ` and `II:MIN`). Chords may also retain the same function, if they share a number of tones (e.g. `V:MAJ` and

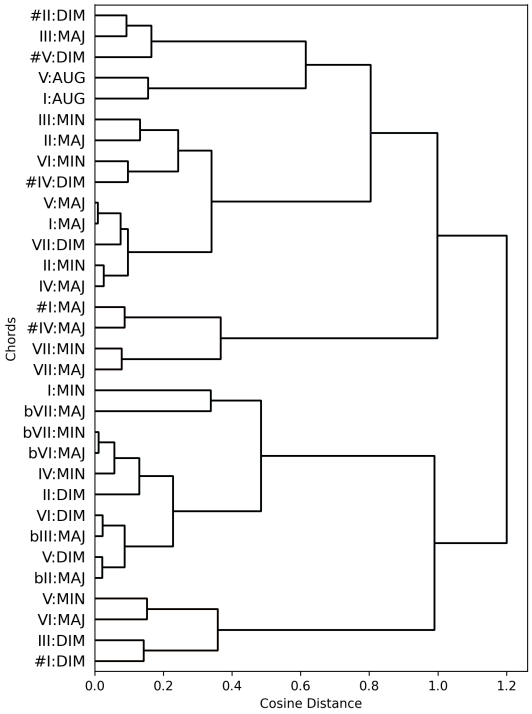


Figure 3. Dendrogram for chord embeddings in major.

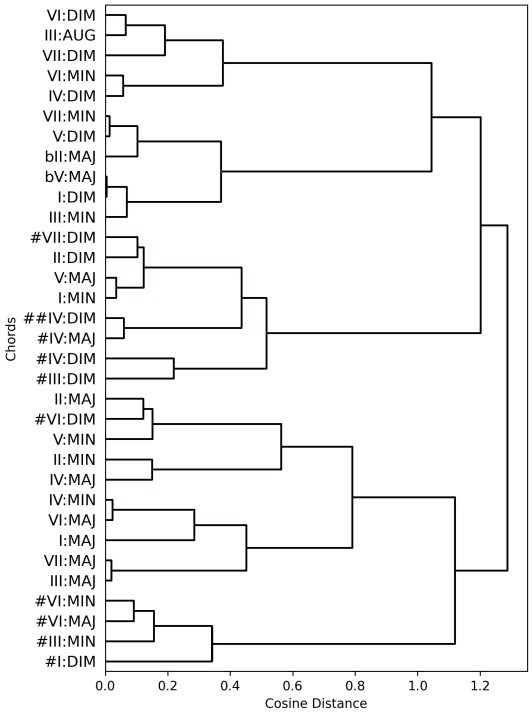


Figure 4. Dendrogram for chord embeddings in minor.

#VII:DIM jointly form a dominant seventh chord in any minor key).

In the major mode (Figure 3), the relative chords that are clustered together are II:MIN and IV:MAJ as well as IV:MIN and bVI:MAJ. The parallel chords are VII:MAJ and VII:MIN, and the chords involved in other common-tone relations are V:MAJ and VII:DIM; II:MAJ and #IV:DIM; II:DIM and IV:MIN; III:MAJ and #V:DIM; VI:MIN and #IV:DIM as well as III:DIM and #I:DIM.

In minor (Figure 4), the relative chords close to one another in the embedding space are VII:MIN and bII:MAJ; IV:MAJ and II:MIN; as well as IV:MIN and VI:MAJ. The parallel chords in minor are #VI:MAJ and #VI:MIN; and, finally, the chords with other common-tone relations are II:DIM and #VII:DIM; I:DIM and III:MIN; V:DIM and VII:MIN; #III:MIN and #I:DIM; as well as #IV:MAJ and ##IV:DIM.

Overall, in our chord embeddings, the relative and common-tone relations are much more frequent than parallel relations, which is to be expected, since the latter involves a change of mode and the sections from which the chords are drawn are precisely defined as staying within one mode (major or minor, notwithstanding potential singular exceptions).

5.1.2 Functional difference

Chords are functionally different if they, or their equivalents, are separated by a perfect fifth, as for example in tonic-dominant or tonic-subdominant pairs, e.g. in authentic or plagal progressions. Note, however, that pairs of chords in the embedding space are undirected. In the major mode (Figure 3), we find fifth-based relations between chords in the embedding space for I:MAJ and V:MAJ; I:AUG and V:AUG; III:MAJ and #II:DIM;[1] as well as #IV:MAJ and #I:MAJ. In the minor mode (Figure 4), we find I:MIN and V:MAJ; II:MAJ and V:MIN; #VI:MAJ/MIN and #III:MIN; I:MAJ and IV:MIN; as well as III:MAJ and VII:MAJ

It is notable that the main cadential chords in both modes (i.e. triads on scale degrees I, V, IV, II, and VII in major, and I, V, and II in minor) occur in relatively close proximity. Despite the fact that distances in embedding spaces are generally hard to interpret, we take the ubiquity of relative, parallel, subset, and fifths-based relations to be an indicator for their pervasiveness in the harmonic progressions in our corpus.

5.2 Chord prediction indicates historical differences in harmonic styles

Here, we summarise the results obtained in chord prediction. Since a composer's prediction accuracy may change for each run of our algorithm due to random initialization of the Word2Vec and LSTM models, we run each experiment ten times, and report mean and standard deviation values for each composer. These are plotted in Figure 5, per composer and mode, where each point represents the mean accuracy for all chords combined, and the shaded bands show the standard deviation across the ten runs. The composers are ordered by their year of death in order to investigate historical trends.

The first thing to notice is that the standard deviations are all quite small (< 0.04 in all cases), showing that our results are consistent across runs and are not affected by random noise in the modeling process. Furthermore, the approximately "inverted U-shape" of the mean values implies that Classical composers are the most predictable from our data, followed by Baroque and Romantic composers, with Modernist and Renaissance composers being the least predictable. This is not to say that Classical composers are more predictable *in general* than composers from other eras. Indeed, remembering that for each composer we train on the data from all other composers in the corpus, this trend is roughly implied by the distribution of data shown in Figure 1. However, the very fact that such an effect exists suggests that composers of the different eras do use chords in fundamentally different ways. Since each model is trained on a very similar set of data (differing by only one composer), the learned model is necessarily similar across composers. Therefore, if two composers used chords similarly, their results would likewise be extremely similar. So, the fact that we see a historical trend *at all* suggests that composers of the different eras do indeed use chords in fundamentally different ways (although we make no claim here about what those differences are).

Furthermore, since the majority of our data comes from Classical composers, we can hypothesize that the mean accuracy of a composer should be positively correlated with the similarity of that composer's chord usage to that of an average Classical composer. From this perspective, the overall shape of the curve makes a lot of sense.

An analysis of the detailed per-chord accuracy results (data available with the code), gives even more insight about the idioms common to a specific composer or period. The strongest result, in a major context, is the very low prediction accuracy for I:MAJ and V:MAJ (the easiest chords to predict overall) when testing on Ravel and Debussy. Indeed, they are two Impressionist composers, who are generally known for their "distinct" harmonies, which rarely (if ever) use authentic cadences. Moreover, we find IV:MAJ and II:MIN to be two "polarising" chords: for most composers, we either predict them very well or very poorly compared to the average. In particular, IV:MAJ is only well predictable for Baroque composers, while others (with the exception of Beethoven, Chopin, and Dvořak) seem to use it in a more peculiar way. II:MIN, on the other hand, only becomes hard to predict from the late Romantic period. This latter result, albeit neat and striking, is not as easily interpretable as the previous one. In minor sections, a low accuracy on I:MIN (the most common chord together with V:MAJ) for Renaissance composers (Gesualdo, Sweelinck, Monteverdi, Schütz) and for Modernists, again signals that this chord has played diverse roles across the centuries. We achieve a relatively low accuracy on many of the most common minor chords for both Romantic and Modernist composers, with the exception

[1] We interpret #II:DIM as a shortened VII:DOM7.

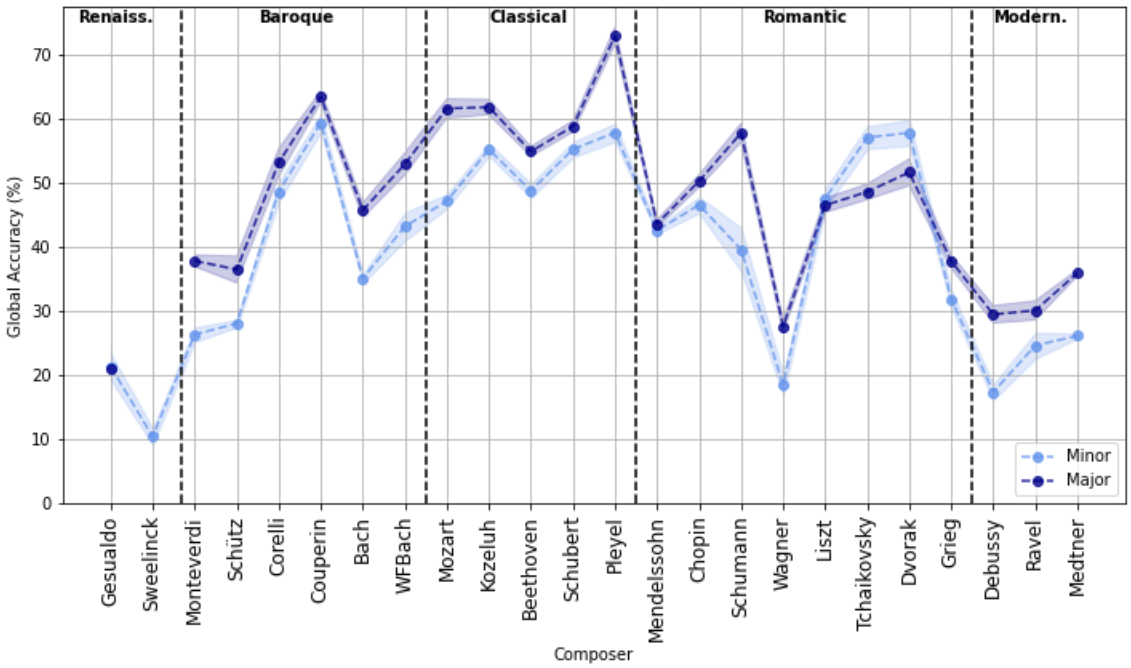


Figure 5. Global chord prediction accuracy for each composer, for major and minor sections. Standard deviation is given by the shaded region around each point. Composers are ordered chronologically by year of death.

of Tchaikovsky. This indicates that he is closer to Classical composers in his works in minor contexts (indeed, his only work in the dataset are the *Seasons*, a collection of rather traditional piano pieces overall). Changes in chord predictability related to stylistic differences are supported by historical studies focusing on the pitch-class content of musical pieces [25, 26].

6. CONCLUSIONS AND FUTURE WORK

In this paper, we investigated progressions of chords in both the major and minor mode by a number of different composers. Our study explored two applications of deep learning methods to music theory: which inferences about tonal relations between chords could be drawn from embedding them in a lower-dimensional space, and whether attempting to predict chords based on the regularities in the data would reveal stylistic differences between composers across historical periods. All data and code are available at `https://github.com/DCMLab/chordembeddings-smc2021`.

Word2Vec was our first processing step, which provided useful grounds to base our subsequent analyses on. When applied to the output vectors of Word2Vec, clustering could capture some well-known tonal relationships between chords, including relative, parallel, and subset relations, as well as (possibly transposed) tonic-dominant pairs of chords. On the other hand, LSTM-based chord prediction yielded fairly high accuracy results in general (roughly 50% for most composers), but it also allowed us to use their high variability across chords and composers to draw some conclusions about chord usage across time which are supported by music theory. Globally, we found that Classical and Baroque composers use chords in a similar way, while Modernists and Renaissance composers seem to have a more distinctive style. The Romantic style seems to be complex, as there is a high variance in how composers from that era use chords.

Future work might also include a more refined use of clustering, for instance by applying it to a Word2Vec model trained only on a single composer—or on a group of composers which are known to be relatively similar to each other—in order to detect some special tonal relationship unique to that set of composers. Alternatively, chord prediction could be employed to investigate how rigidly a given composer belongs to a given artistic era: by restricting the training corpus to composers in the same era, we would prevent the model from learning totally unrelated idioms, thus achieving a higher accuracy on the test composer (to an extent depending on how similar he actually is to the others in that era).

As mentioned, in the current work, we identified the existence of historical differences in chord usage. However, we did not identify what those differences were. Future work could look at the problem from a more causal perspective by limiting the training corpus for each composer to only those composers who preceded them.

Acknowledgments

Research supported through the Swiss National Science Foundation within the project "Distant Listening – The Development of Harmony over Three Centuries (1700–2000)" (Grant no. 182811).

7. REFERENCES

[1] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013, pp. 3111–3119.

[2] D. Herremans and C.-H. Chuan, "Modeling musical context with word2vec," in *Proceedings of the First International Conference on Deep Learning and Music*, 2017, pp. 11–18.

[3] C.-H. Chuan, K. Agres, and D. Herremans, "From context to concept: exploring semantic relationships in music with word2vec," *Neural Computing and Applications*, vol. 32, no. 4, pp. 1023–1036, 2020.

[4] S. Madjiheurem, L. Qu, and C. Walder, "Chord2vec: Learning musical chord embeddings," in *Proceedings of the constructive machine learning workshop at 30th conference on neural information processing systems (NIPS2016), Barcelona, Spain*, 2016.

[5] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86–97, 2012.

[6] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979. [Online]. Available: http://www.jstor.org/stable/2346830

[7] F. Korzeniowski and G. Widmer, "Improved chord recognition by combining duration and harmonic language models," in *ISMIR*, 2018.

[8] E. J. Humphrey, T. Cho, and J. P. Bello, "Learning a robust tonnetz-space transform for automatic chord recognition," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 453–456.

[9] B. Duane and J. Jakubowski, "Harmonic clusters and tonal cadences: Bayesian learning without chord identification," *Journal of New Music Research*, vol. 47, no. 2, pp. 143–165, 2018.

[10] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, pp. 2451–71, 10 2000.

[11] A. Ycart, A. McLeod, E. Benetos, and K. Yoshii, "Blending acoustic and language model predictions for automatic music transcription," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2019, pp. 454–461.

[12] K. Landsnes, L. Mehrabyan, V. Wiklund, R. Lieck, F. C. Moss, and M. Rohrmeier, "A model comparison for chord prediction on the Annotated Beethoven Corpus," in *Proceedings of the 16th Sound & Music Computing Conference. Málaga, Spain*, 2019.

[13] L. Feisthauer, L. Bigo, and M. Giraud, "Modeling and learning structural breaks in sonata forms," in *ISMIR*, 2019.

[14] M. Neuwirth, D. Harasim, F. C. Moss, and M. Rohrmeier, "The Annotated Beethoven Corpus (ABC): A dataset of harmonic analyses of all Beethoven string quartets," *Frontiers in Digital Humanities*, vol. 5, p. 16, 2018.

[15] F. C. Moss, M. Neuwirth, D. Harasim, and M. Rohrmeier, "Statistical characteristics of tonal harmony: A corpus study of beethoven's string quartets," *PLOS ONE*, vol. 14, no. 6, pp. 1–16, 06 2019. [Online]. Available: https://doi.org/10.1371/journal.pone.0217242

[16] F. C. Moss, "Transitions of tonality: A model-based corpus study," Ph.D. dissertation, EPFL, 2019.

[17] J. Hentschel, M. Neuwirth, and M. Rohrmeier, "The Annotated Mozart Sonatas: Score, Harmony, and Cadence," *Transactions of the International Society for Music Information Retrieval*, vol. 4, no. 1, pp. 1–14, 2021.

[18] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[19] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980 [cs]*, 2017.

[20] Z. Gárdonyi and H. Nordhoff, *Harmonik*. Wolfenbüttel: Möseler Verlag, 2002.

[21] F. C. Moss, "Tonality and functional equivalence: A multi-level model for the cognition of triadic progressions in 19th century music," in *International Conference of Students of Systematic Musicology - Proceedings*, vol. 1, London, 2014, pp. 1–8.

[22] M. Rohrmeier, "The Syntax of Jazz Harmony: Diatonic Tonality, Phrase Structure, and Form," *Music Theory and Analysis (MTA)*, vol. 7, no. 1, pp. 1–63, Apr. 2020.

[23] M. Rohrmeier and I. Cross, "Statistical Properties of Tonal Harmony in Bach's Chorales," in *Proceedings of the 10th International Conference on Music Perception and Cognition*, 2008, pp. 619–627.

[24] H. Riemann, *Vereinfachte Harmonielehre oder die Lehre von den tonalen Funktionen der Akkorde*. London: Augener, 1893.

[25] C. Weiß, M. Mauch, S. Dixon, and M. Müller, "Investigating style evolution of Western classical music: A computational approach," *Musicae Scientiae*, vol. 23, no. 4, pp. 486–507, 2019.

[26] D. Harasim, F. C. Moss, M. Ramirez, and M. Rohrmeier, "Exploring the foundations of tonality: Statistical cognitive modeling of modes in the history of Western classical music," *Humanities and Social Sciences Communications*, vol. 8, no. 1, 2021.

ESTIMATING UNEXPECTEDNESS IN JAZZ HARMONY WITH A PROBABILISTIC INCREMENTAL PARSER

Yuta OGURA (小椋裕太)[1,2], **Hidefumi OHMURA** (大村英史)[2], **Satoshi TOJO** (東条敏)[3], and **Kouichi KATSURADA** (桂田浩一)[2]

[1]**Yamaha Corporation**, 10-1 Nakazawa-cho, Naka-ku, Hamamatsu, Shizuoka, 430-8650, Japan
[2]*Department of Information Sciences*, **Tokyo University of Science**, 2641 Yamazaki, Noda, Chiba, 278-8510, Japan
[3]*Graduate School of Information Science*, **Japan Advanced Institute of Science and Technology (JAIST)**, 1-1 Asahidai, Nomi, Ishikawa, 923-1292, Japan

ABSTRACT

Cognitive music theory analyzes the listeners' understanding of music. The generative syntax model (GSM) has shown that the structure of expectation-realization in a harmonic progression becomes recursive and hierarchical, in terms of context-free grammars. However, GSM only takes into consideration the cognitive structure after listening and does not discuss the dynamic process during listening, but given that music is a temporal structure of sound, dynamic changes in cognitive structure are more important. In this study, we extend the GSM by using probabilistic context-free grammar to represent the cognitive structure for each successive chord. Furthermore, we implemented a harmonic analysis system based on the extended model. We use a jazz standard, a genre of music in which harmonic progression is particularly important, as a case study, analyze it, and show its efficacy. The experimental result quantified its unexpectedness, appearing in the middle of a piece of music.

1. INTRODUCTION

The origin of music is said to be closely related to the evolution of language [1], and thus, "what is music?" is a historically abstruse question. The first theorized music seems Pythagoras' pitch in ancient Greece, however, along with the history the music has diversified into various genres, and in accordance with the theories also have been complicated. On the other hand, many fundamental questions, such as "how do we understand music?" still remain unclear. Cognitive music theory focuses on such questions. Cognitive music theory analyzes music based on the cognitive processes of the "listener", whereas a general music theory is used as a tool for the music "creator", that is, to compose and arrange music [2, 3].

In cognitive music theory, there is a method of analyzing music as a hierarchical structure. The notion of a hierarchical structure of a piece of music originated from the reduction hypothesis proposed by Schenker [4]. The reduction hypothesis states that "a listener of a piece of music will try to organize all pitch events (notes and chords) into a hierarchical structure of relative importance." Lerdahl and Jackendoff's *A generative theory of tonal music* (GTTM) [5] analyzes the melody of a piece as a hierarchical structure [6–8]. *The generative syntax model* (GSM) by Rohrmeier focuses on harmony and defines context-free rules for harmonic progressions [9]. This allows the hierarchical structure of harmonic cognition to be represented as a tree structure, as shown in Figure 1.

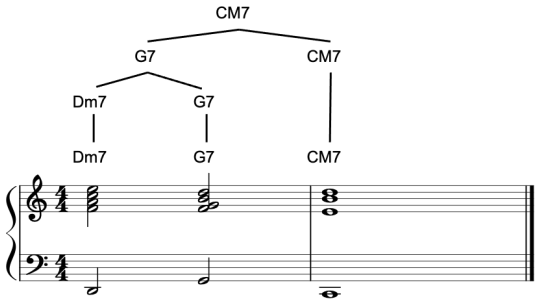


Figure 1. Structural analysis of ii-V-I based on GSM

However, GSM only takes into consideration the cognitive structure after listening and does not discuss the dynamic process during listening, but given that music is a temporal structure of sound, dynamic changes in cognitive structure are more important. The philosopher Meyer states that "the meaning of music arises from the relation of sounds in which the preceding sound somehow **expects** the following sound, and the embodiment of the following sound tries to confirm or review the preceding sound [10]." The interest in music is formed by incremental cognition.

Furthermore, in the original GSM, we cannot compare the tree structure if multiple analyses are probable due to generative syntax. This is because there is no concept of probability. The degree of expectation in the middle of a piece can be expressed by probabilities.

In this study, we focused on the incremental cognition of music. To clarify the cognitive structure for each successive chord, we propose the application of incremental chart parsing [11] to GSM. In addition, we extend the grammat-

ical rules of GSM to probabilistic context-free grammar, to enable a quantitative discussion of unexpectedness in a harmonic progression. This makes it possible to compare the importance of different tree structures. The proposed method is implemented on a computer, and incremental analysis is performed on a jazz piece to discuss where the unexpectedness occurs within the music.

This paper is organized as follows: In the following section, we summarize the theory of GSM; In section 3, we detail the mechanism of incremental parser and probabilistic context-free grammar; In section 4, we propose a method for the evaluation of unexpectedness; In section 5, we show an example of incremental analysis with a jazz chord sequence and discuss the unexpectedness; In section 6, we summarize our contributions.

2. GENERATIVE SYNTAX MODEL

2.1 Overview

The GSM [9, 12] is a cognitive music theory proposed by Rohrmeier. GSM is a model that represents the cognitive structure of a musical piece as a tree structure, similar to GTTM [5], a well-known cognitive music theory. Whereas GTTM proceeds without explicit context-free rules, GSM is strongly based on Chomsky's generative grammar theory [13–15], and proceeds with explicit context-free rules for harmonic progressions.

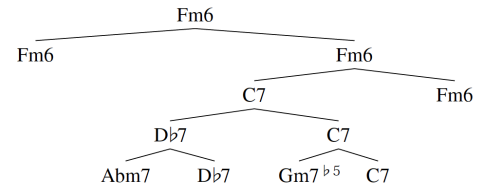
GSM makes the following assumptions about harmonic cognition: one chord has a dependency relationship with the chords before and after it. In particular, an adjacent chord has a "functional head," in which the dominant chord governs a broader time interval absorbing surrounding pitch events.

There are several versions of the phrase structure rules presented in the GSM, depending on the type of music. In the following, we will focus on the rule [12, 16] proposed for jazz music, which is the subject of this study.

All syntactic rules presented in the GSM are said to follow either the **Prolongation principle** or the **Preparation principle** . Figure 2 shows the GSM analysis of the jazz standard Birk's Works (Fm6 A♭m7 D♭7 $\mathrm{Gm7}^{\flat 5}$ C7 Fm6) In the following, We explain the principle of Prolongation and Preparation using this analysis.

The initial Fm6 established the tonic and as such creates the expectation that the progression ends with Fm6. The chords A♭m7 and D♭7 function as the tritone-substituted subdominant and dominant of C7, respectively. They therefore create expectation that resolves in the (temporally distant) chord C7. $\mathrm{Gm7}^{\flat 5}$ can be thought of as a subdominant chord in the F minor key. It herefore creates expectation that resolves with the dominant chord C7 wich itself resolves into the last tonic chord Fm6. We say that the tonic chords constitute a **Prolongation**. The subdominant chords **Prepare** the dominant chords and the dominant chords **Prepare** the tonic chord. Abstractry, we say that a chord X *refers* to a chord Y if X either prolongs or prepares Y.

Figure 2b illustrates the structure of expectation realization in the harmonic progression. Chord pairs, represented by arrows in the diagram, are based on the preparation principle, in the sense that the former chord serves as a preparatory function for the following chord, while pairs such as Fm6 - Fm6, without arrows, are based on the prolongation principle. This is in one-to-one correspondence with the tree structure in Figure 2a, so it can be said that the tree structure of harmonic progressions reveals the structure of expectation–realization in harmonic progressions.



(a) Harmonic syntax tree based on GSM

(b) Harmonic expectation–realization structure

Figure 2. Syntax tree example about the final chords of the jazz standard Birks's Works [16]

There are also two types of prolongation principles: strong prolongation and weak prolongation. A strong prolongation is an extension with the same chord type of the same root, while a weak prolongation is an extension of a chord with the same function (e.g., prolongation by C and Am in the key of C major). A strong prolongation is represented as $X \to X\ X$ for any chord symbol X (e.g., Fm6 $\to$ Fm6 Fm6). Weak prolongation is represented as $X \to Y\ X$ or $X \to X\ Y$ with respect to functionally equal chord symbols X, Y (e.g., Fm6 $\to$ A♭ Fm6). The preparation principle is represented by $X \to Y\ X$ for chord symbols X, Y that are not functionally equal (e.g., Fm6 $\to$ C7 Fm6).

The prolongation and preparation principles can be summarized as follows:

$$\begin{aligned} \textbf{Strong Prol.}\ & X \to X\ X \\ \textbf{Weak Prol.}\ & X \to Y\ X \mid X\ Y \\ \textbf{Preparation}\ & X \to Y\ X \end{aligned}$$

These syntactic rules are characterized by the fact that nonterminal symbols do not have their own category but are expressed in the form of a binary tree, where the left-hand side symbol always appears on the right-hand side. This feature can also be seen in grammatical theories such as dependency grammar [17] and combinatorial category grammar (CCG) [18]. The symbol appearing on both sides of the arrow is called the **head**. In the setting of GSM, the prolonged (resp. prepared) chrod is the head. Therefore, the preparation rule is always right-headed, but the weak prolongation rule can be left-headed or right-headed de-

Algorithm 1 Algorithm of incremental parsing

function CHART_PARSING(G_chart, w)
  $L\_chart \leftarrow \{\}$ *Local charts*\
  $temp \leftarrow \{\}$

  step1 Lexicon Consultation\
  for $\alpha \in$ Lexicon **do**
    if $w = \alpha$ **then**
      $L\_chart \leftarrow L\_chart \cup \{[w]_\alpha\}$

  step2 Rule Application\
  for $\sigma \in L\_chart$ **and** $\beta \rightarrow \beta_1\beta_2 \ldots \beta_n \in$ Rules **do**
    if $\sigma = \beta_1$ **then**
      $L\_chart \leftarrow$
       $L\_chart \cup \{[\sigma[?]_{\beta_2}]_\beta\}$

  step3 Term Replacement\
  for $\phi \in G\_chart$ **and** $\psi \in L\_chart$ **do**
    if $\gamma = lut(\phi) \wedge \gamma = \psi$ **then**
      replace $lut(\phi)$ with ψ
      $temp \leftarrow temp \cup \{\phi\}$
  $G\_chart \leftarrow temp$ *Global charts*\
  return G_chart

main\
$G\_chart \leftarrow [?]_S$ *initialize*\
for i=1,...,last **do**
  $w_i \leftarrow$ input_chord
  $G\_chart \leftarrow$ CHART_PARSING(G_chart, w_i)

pending on the interpretation.

2.2 Jazz Harmony Treebank

The Jazz Harmony Treebank (JHT)[1] [16] is a dataset annotated with the results of the hierarchical analysis of harmonic progressions in jazz standards by experts. Hierarchical analysis was based on the aforementioned GSM principles. In this study, it was used as a corpus to estimate the probability of applying the probabilistic context-free grammar described below.

The analysis of JHT is based on 150 jazz tunes in the genres of Swing, Bossa Nova, Jazz Blues, Bebop, Cool Jazz, and Hard Bop, and does not include non-tonal genres such as Modal Jazz, Free Jazz, and Modern Jazz.

3. INCREMENTAL STRUCTURAL ANALYSIS

3.1 Incremental Chart Parsing

We have proposed a model that displays the tree structure, for each successive chord, by incrementally analyzing harmonic progressions [19]. This model was realized by applying a natural language parsing method, incremental chart parsing [11], to the GSM. Here, we explain this algorithm, which is a natural language processing technique, we use the word "word" to describe it, but in harmony, "word" refers to a chord symbol (e.g., CM7, G7, etc.).

[1] https://github.com/DCMLab/JazzHarmonyTreebank

In natural language processing, the input is a sequence of words spaced by blanks. Each word is positioned by numbers, called *nodes*, placed at the blanks between words; thus, word w_i resides between node $i-1$ and node i.

An *edge* combines one node with another. A tree is represented by a data structure, called *term*; when α belongs to category X, we write it as $[\alpha]_X$. Here, α is either a word (chord), a term, or a list of terms. A *chart* consists of an edge and term. For example, when a chart is (i, j) and $[[\alpha]_Y[\beta]_Z]_X$, it represents a (local) tree obtained by an application of production rule '$X \rightarrow Y\ Z$' between nodes i and j to the sequence of $\alpha\beta$, being recognized by α and β belonging to Y and Z, respectively. In contrast, an edge can possess multiple terms; that is, there may be multiple parse trees on the edge. Thus, there might be different interpretations of edges. The term displayed by $[?]_X$ is called an undecided term, where the content of category X is not decided. When an undecided term resides on an edge, the edge is called *active*; otherwise, *inactive*.

In incremental chart analysis, when the i-th word w_i is input, the following operations are performed sequentially:

Lexicon Consultation When the category of w_i is X, add an inactive edge labeled by term $[w_i]_X$ on $(i-1, i)$.

Rule Application When there exists an active edge labeled by term $[\ldots]_X$ on $(i-1, i)$, for all grammar rules such as $A \rightarrow XY \ldots Z$, add an edge labeled by term $[[\ldots]_X[?]_Y \ldots [?]_Z]_A$ on $(i-1, i)$.

Term Replacement Let ϕ, ψ be terms, and $[?]_X$ be the leftmost undecided term of ϕ labeled on $(0, i-1)$. If the category of ψ labeled on $(i-1, i)$ is X, add an edge labeled by a term that replaces the leftmost undecided term of ϕ with ψ to $(0, i)$.

Algorithm 1 shows the above operations in pseudo-code. In pseudo-code, an edge is represented by a pair of indices in the array; hence, an edge is not mentioned explicitly. Furthermore, σ, ψ, ϕ, and γ represent terms, and when terms are connected by equals ($=$), it indicates that the outermost categories are equal. In addition, we denote the left-most undecided term of term ϕ as $lut(\phi)$.

In general, chart parsing takes the whole sentence as an input and constructs a tree. On the other hand, in incremental chart parsing, parts of a sentence are input sequentially, and the tree is constructed incrementally. There are two types of algorithms for chart analysis: **bottom-up** and **top-down**. The bottom-up algorithm starts with a word and builds a tree toward the start symbol S. The top-down algorithm starts with the start symbol S and builds a tree toward the leaves, that is, the word. A combination of bottom-up and top-down algorithms can be used to deal with sequential inputs. Therefore, incremental chart parsing introduces two top-down operations into the bottom-up chart analysis, namely, the operation of applying a grammar rule to an active arc and the operation of replacing the leftmost undecided term, of a term labeled with an active arc, with a term labeled with another active arc. In the actual system, only the global chart at each stage is displayed. In this study, we refer to these terms as **candidate trees**.

In addition, it is necessary to initialize the global chart with an undecided term whose category is the start symbol S. In this paper, following tonic chord for all 12 keys are used for start symbol S.

$$\mathrm{S} = \{\mathrm{C}, \mathrm{D}\flat, \mathrm{D}, \ldots, \mathrm{B}\} \times \{\mathrm{M}, \mathrm{M7}, \mathrm{m}, \mathrm{m7}\} \quad (1)$$

3.2 Probabilistic Context Free Grammar

Probabilistic context-free grammar (PCFG) models extend context-free grammars and can calculate the probability of occurrence of a syntax tree [20]. This model assigns the following conditional application probabilities to each generative rule in the grammar $\mathrm{A} \rightarrow \alpha$.

$$P(\mathrm{A} \rightarrow \alpha | \mathrm{A}) \quad (2)$$

Since it is a conditional probability, the following equation holds.

$$\sum_{\alpha} P(\mathrm{A} \rightarrow \alpha | \mathrm{A}) = 1 \quad (3)$$

In other words, the sum of the probabilities of applying the generative rules, with the same non-terminal symbol (pre-terminal symbol[2]) on the left side was 1. The simplest way to calculate such a probability is to use a parsed corpus. The probability of applying the generative rule can be calculated as follows:

$$P(\mathrm{A} \rightarrow \alpha | \mathrm{A}) = \frac{\text{Number of A} \rightarrow \alpha \text{ in the corpus}}{\text{Number of A in the corpus}} \quad (4)$$

In Equation (4), the denominator is the number of occurrences of non-terminal symbols in the corpus, and numerator is the number of times the generative rule is used.

Obviously, expression 3 is satisfied. In addition, given the application probabilities in this way, the generation probability of a certain tree structure t can be given by the product of the application probabilities of all the generative rules that make up the tree structure.

To prevent the exponential increase in analysis time with longer sentences in the incremental chart analysis, we performed branch trimming, using the generation probability of the tree structure at each word stage. The terms stored in the global chart, at the time of each word, up to the top 100 terms in probability, were retained for the analysis of the next word.

3.3 Expectation-based Chord Sequence Analyzer

In this study, we implemented a GUI application called **expectation-based chord sequence analyzer** (ECSA)[3] . The main purpose of this application is to intuitively understand the harmonic structure.

Figure 3 shows ECSA's main view. When we enter a chord sequences in the text box on the top page, the results of the tree structure analysis for the input are displayed in multiple lines. Each line shows the result of the analysis up to the point when the chord was entered. Figure 3 is the analysis result of jazz standard *Cute*, which will be explained in section 5. The first line shows the results of the analysis at the stage when chords up until Dm7 are input, and the second line shows the results when up to when Dm7 and G7 are input. Each parse tree, at that time, is displayed in a slider (carousel) panel. At the top of each panel, the generation probability of the parse tree is shown, and the parse trees are sorted from left to right with the highest probability. Also, the number next to the label of each chord name shows the unexpectedness measure U that will be described in section 4.

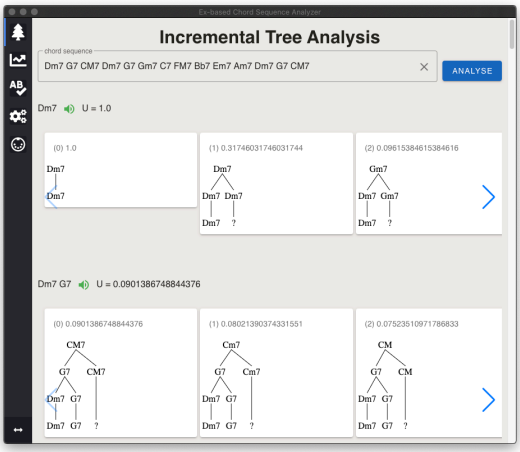


Figure 3. Appearance of Expectation-based Chord Sequence Analyzer

4. EVALUATION OF UNEXPECTEDNESS

In this study, unexpectedness is considered to arise from expectation–realization and expectation–deviation. A harmonic progression with "low" unexpectedness is a harmonic progression in which the expectation of the preceding chord is realized by the following chord. A harmonic progression with "high" unexpectedness is a harmonic progression in which the chord deviates from the expectation set up by the preceding chord.

This expectation–realization and expectation–deviation depends on the growth process of the tree. In a harmonic syntax tree in the middle of a piece of music, the chords expected to follow are represented as categories of undecided terms such as $[?]_{\mathrm{CM7}}$. In the next step, we consider an expectation–realization to have occurred when chord CM7 is actually input, and an expectation-deviation to have occurred when another chord is input. In the following, we refer to the stage in the middle of a piece, where a certain chord is input as the **chord step**.

The change in the generation probability of the tree structure is also important for unexpectedness. In general, there are multiple candidate trees for each chord step, and the

[2] This is the equivalent of phrases such as NP and VP in natural language processing. In this study, we follow the example of [12, 16] and use a grammar rule that equates non-terminal with pre-terminal symbols, namely there are no lexicons.

[3] https://github.com/yutaogura/Ex-based-Analyzer

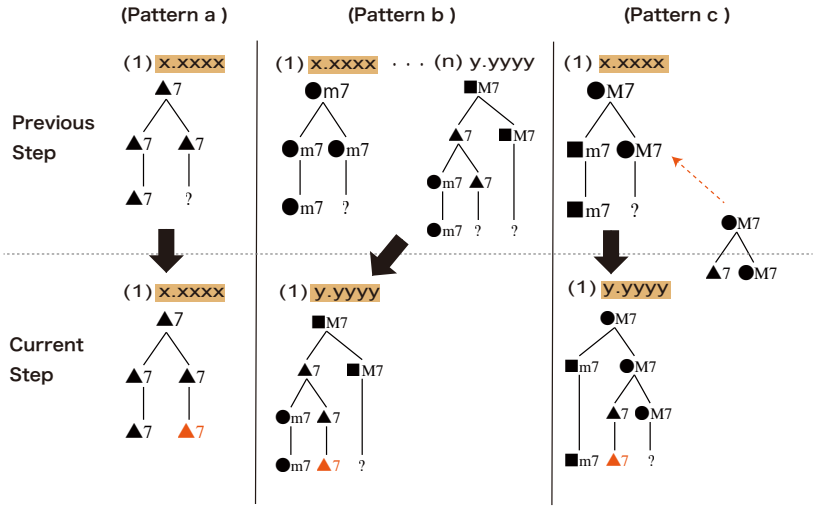


Figure 4. Tree structure change pattern with maximum generation probability

candidate trees are ranked by their generation probabilities. In this study, we focus on the tree structure with the highest generation probability, at each chord step, as a representative of the cognitive structure at each chord step.

Figure 4 shows a schematic diagram of how the tree structure, with the highest generation probability, changes at a given chord step. The numbers in parentheses indicate the rank of the probability at that chord step. In the following, the tree structure from which a certain tree structure is grown is called the derivation tree structure. The thick solid arrows show the relationship between the source and destination of the tree.

Pattern a and **b** in Figure 4 show how the chord expected in the previous chord step is realized in the current step. In **Pattern a**, there is no change in the value of the probability of generating the tree with the highest probability, but in **Pattern b**, the value of the probability of generating the tree with the highest probability has changed because the rank of the tree structure that was n-th in the previous step has become the first. **Pattern c** shows the input of a chord that was not expected in the previous step. In this case, a new substructure is added (dotted arrow in the figure), and the value of the probability of generating the tree with the highest probability changes.

Based on the above discussion, we formulate a measure of unexpectedness that considers the degree of increase in rank and the addition of substructures. Let $t^{(n)}$ denote the candidate tree structure at a certain chord step n, and let $t^{(n)}_{\mathrm{maxprob}}$ denote the tree structure with the highest generation probability, and $P(t^{(n)}_{\mathrm{maxprob}})$ denote the probability value. The tree structure from which $t^{(n)}$ is derived is denoted as $t^{(n-1)}$. In this case, the unexpectedness $U^{(n)}$ of a chord step n is given as follows:

$$t^{*} = t^{(n)}_{\mathrm{maxprob}} \tag{5}$$

$$A = \frac{P\left(t^{*(n-1)}\right)}{P\left(t^{(n-1)}_{\mathrm{maxprob}}\right)} \tag{6}$$

$$B = \frac{P\left(t^{(n)}_{\mathrm{maxprob}}\right)}{P\left(t^{*(n-1)}\right)} \tag{7}$$

$$U^{(n)} = \begin{cases} P\left(t^{(n)}_{\mathrm{maxprob}}\right) & (n=1) \\ A \times B = \frac{P\left(t^{(n)}_{\mathrm{maxprob}}\right)}{P\left(t^{(n-1)}_{\mathrm{maxprob}}\right)} & (n>1) \end{cases} \tag{8}$$

The closer the measure of unexpectedness U is to 1, the more the expectation–realization has occurred, meaning a "low" unexpectedness, and the closer it is to 0, the more the expectation–deviation has occurred, meaning a "high" unexpectedness. The A represents the scarcity of the source tree structure in the previous chord step, that is, the increase in rank (Eq. 6). Also, B represents the probability of generating the newly added substructure (Eq. 7). The unexpectedness measure U is a combination of these. In the actual calculation, the $P(t^{*(n-1)})$ parts cancel each other out, so in the end, U is just the ratio of the probability of generating the tree with the highest probability before and after the target chord step.

5. CASE STUDY WITH A JAZZ CHORD SEQUENCE

In this study, we present an example of incremental structural analysis using an actual jazz standard *Cute*.

Cute consists of 32-bars ABAC form. In this section, we analyze the AC part, which is the second half of the 16-bars. The chord progression and lead melody are shown in Figure 5.

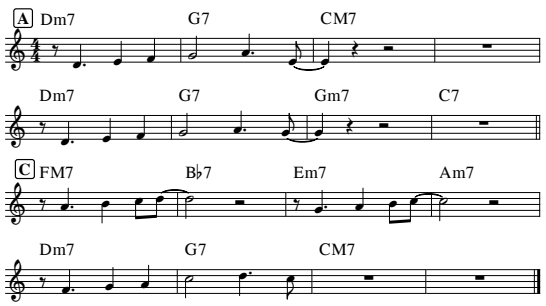


Figure 5. The chord progression and lead melody of *Cute* (second half 16-bars)

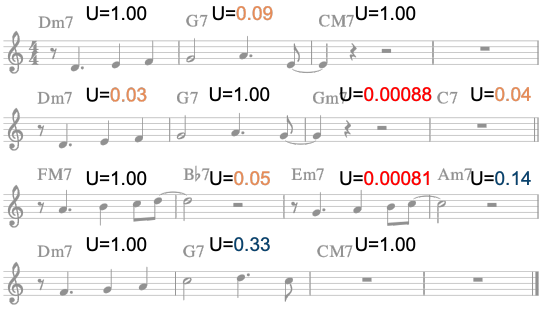


Figure 6. U value at each chord step of *Cute*

Before discussing the unexpectedness of the piece, using incremental structural analysis, let us review the basic characteristics of the piece from the perspective of conventional music theory [21, 22]. *Cute* is in the key of C major. This is evident from the fact that the last chord of the song ends with CM7. The first four measures, Dm7 – G7 – CM7, are two-five-one of the C major key. The following bars 5 and 6 are also two-five-one in the key of C major, and in the seventh bar, chord CM7, which is tonic, is expected to come, but the chord Gm7 is inserted, and from here, Gm7 – C7 – FM7, two-five-one in the key of F major, begins. This is followed by B♭7, a subdominant minor, and Em7, a diatonic chord in the key of C major, and then iii – vi – ii – V – I, leading to tonic CM7.

The results of the tree structure analysis are shown in Figure 7. The figure shows the tree structure output by the system for each chord step. The system can display up to the maximum number of candidate trees for branch trimming at each chord step, but only a few of them are shown for the sake of space limitations. (a)–(h) show the chord steps in the chord progression of *Cute*. In the upper part of each tree structure, the generation probability of the candidate tree and the ranking of the generation probability for each chord step are shown in parentheses. In the following, a candidate tree whose probability of generation is *n*-th in chord step (a) is denoted as (a–n). '?'(question mark) denotes an undecided term and indicates the next expected chord or category. The rank in parentheses with an asterisk (*) indicates an inactive arc, that is, a closed tree structure.

If we look at the growth process of the tree structure in order, we can see that at each chord step, various chords are expected to be realized in the next step, and the tree structure is recombined. Looking at the evolution of the tree structure, up to (d–1), we can see that the tree grows as (a–4)→(b–1)→(c–2)→(d–1). CM7 is the tonic in this piece, and the progression of Dm7 – G7 – CM7 forms a group. Thus, the closed tree structure is considered to be harmonically stable at this point.

Next, the unexpectedness value U is calculated for all the chord steps, as shown in Figure 6. It can be seen that the value of U decreases from the 6th to the 7th bars, which is the part where two-five-one in the key of F Major appears. In the actual tree structure, G7 is expected to be the parent of Gm7 in (f–1), but in (g–1), the insertion of C7 causes a recombination of the tree structure, and FM7 is expected to be the parent.

Then, FM7 at the 9th bar is inserted such that the expectation of 8th bars is realized, and the value of U is lowered again in the following B♭7 and E7. This is thought to be caused by the non-diatonic chord B♭7. In general music theory, B♭7 is considered to be a sub-dominant minor chord. It comes from the iv chord in key of C minor, which is the parallel key[4] . Together with FM7, FM7–B♭7 this progression is famous for the formation of a subdominant-subdominant minor chord progression. In this case, FM7 is often analyzed as working as a pivot chord[5] , while tonally it remains in C major. Therefore, the fact that the value of U is lower in Em7, which is often analyzed as a tonic in C major, should be reconsidered as whether it has cognitive reality[6] or not.

6. CONCLUSION

In this study, we focused on the cognitive structure for each successive chord and proposed an incremental structural analysis of jazz harmony, based on the generative syntax model [9, 12]. Especially, we have employed probabilistic context-free grammar (PCFG) instead of traditional CFG, and thus, we could externalize the unexpectedness U, concerning the growth process of syntactic tree. Through the analysis of jazz music, using the implemented system ECSA, it became possible to quantitatively evaluate the position of unexpectedness in the music.

The importance of expectation-realization in music cognition has been discussed in Narmour's implication-realization model [23], but it was limited to the analysis of an entire piece. The main contribution of this study

[4] A major scale and a minor scale that have the same tonic are called parallel keys. In this case, C major and C minor is parallel key.

[5] A chord that has a function across multiple tonalities. In the case of FM7, one is I (tonic) in the key of F major, another is IV (subdominant) in the key of C major.

[6] When a concept or model can rationally explain a cognitive or psychological phenomenon, it is said that the concept or model has cognitive reality.

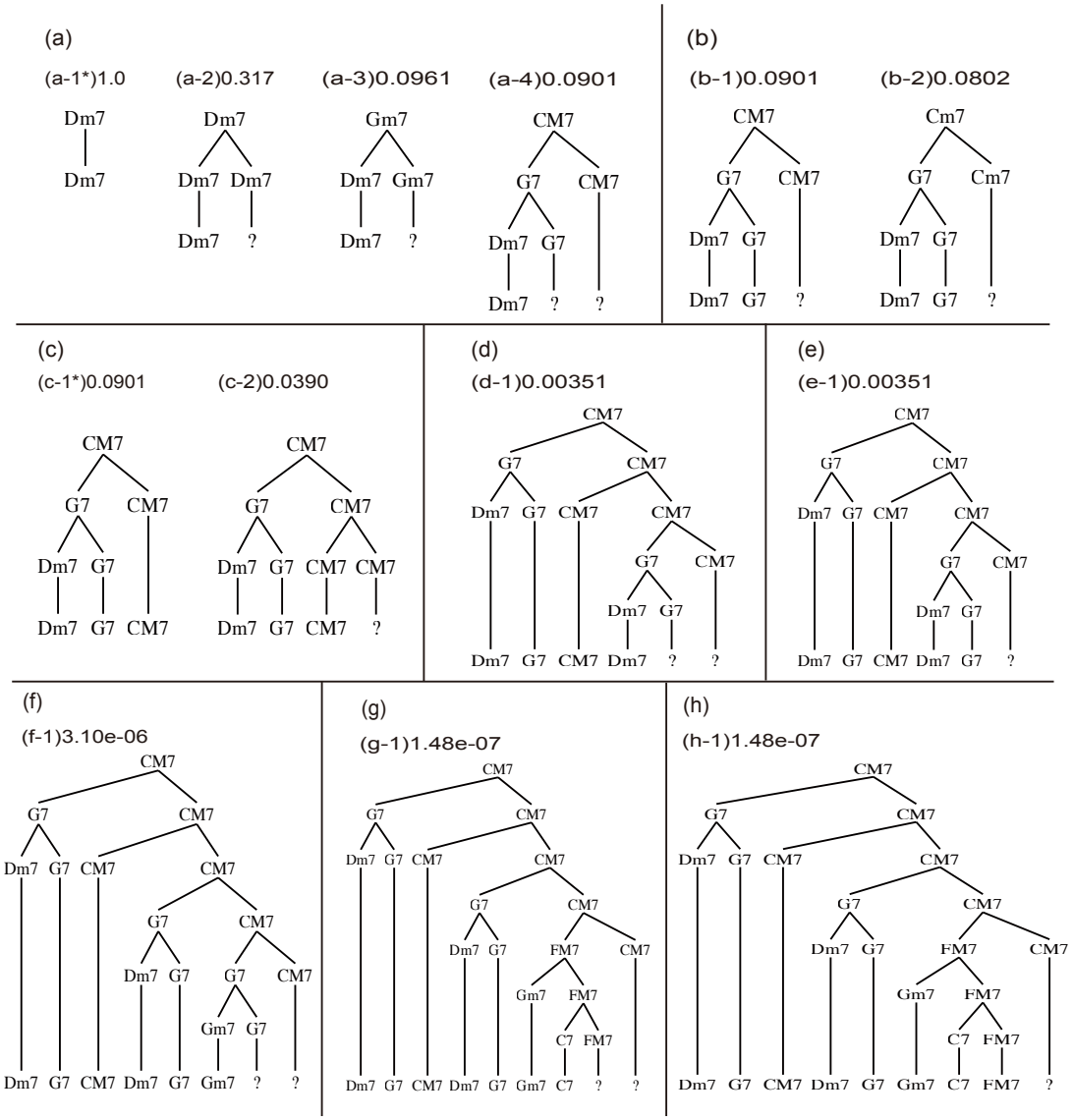


Figure 7. Incremental structural analysis of *Cute*

is to propose an analysis method that is closer to human music cognition by representing expectation-realization or expectation-deviation for each successive chord, using an incremental structural analysis method.

As a future work, we believe there is an advantage in investigating the cognitive reality of the measurement of unexpectedness U by experiment. The reason for this is that there is a difference between the position of a piece of music, that we consider surprising based on conventional music theory, and the position where unexpectedness occurs quantitatively using the measurement of unexpectedness U. The position of unexpectedness in a piece of music is thought to vary greatly depending on the individual's musical experience. Therefore, we need to creat a measure reflecting the cognitive differences between individuals, to go back to the grammatical rules themselves and to examine their rationality.

As a possible application, we are considering incorporating it into real-time applications such as automated session systems, taking advantage of the incremental analysis of the sequential interpretation of music flow.

Acknowledgments

This work was supported by JSPS Kaken 16H01744 and 20K12126.

7. REFERENCES

[1] N. L. Wallin, B. Merker, and S. Brown, *The origins of music*. MIT Press, 2000.

[2] S. Tojo and K. Hirata, *Music · Mathmatics · Language -The horizons of music opened up by information science-*. Kindaikagaku co., 5 2017.

[3] K. Ohgushi, S. Kuwano, S. Namba *et al.*, *Music Perception Handbook: Science that challenges the mysterious perception of music*. Kitaohji Shobou co., 2020.

[4] H. Schenker, *Der Freie Satz. Neue musikalische Theorien und Phantasien, Margada, Liège, Belgium*, 1935.

[5] F. Lerdahl and R. S. Jackendoff, *A Generative Theory of Tonal Music*. The MIT Press, 1996.

[6] S. Tojo, Y. Oka, and M. Nisida, "Analysis of chord progression by HPSG." ACTA Press, 2006, pp. 305–310.

[7] M. Granroth-Wilding and M. Steedman, "A robust parser-interpreter for jazz chord sequences," *New Music Research*, vol. 43, pp. 354–374, 2014. [Online]. Available: http://jazzparser.

[8] T. Fukunari, S. Arn, and S. Tojo, "CCG analyzer with tonal pitch space for non-classical chords." Institute of Electrical and Electronics Engineers Inc., 11 2016, pp. 239–246.

[9] M. Rohrmeier, "Towards a generative syntax of tonal harmony," *Journal of Mathematics and Music*, vol. 5, pp. 35–53, 2011.

[10] L. B. Meyer, "Meaning in music and information theory," *The Journal of Aesthetics and Art Criticism*, vol. 15, no. 4, pp. 412–424, 1957.

[11] S. Matsubara, S. Asai, K. Toyama, and Y. Inagaki, "Chart-based parsing and transfer in incremental spoken language translation," in *Proc. the 4th Natural Language Processing Pacific Rim Symposium*, 1997, pp. 521–524.

[12] M. Rohrmeier, "The syntax of jazz harmony: Diatonic tonality, phrase structure, and form," *Music Theory and Analysis (MTA)*, vol. 7, 2020.

[13] N. Chomsky, "Syntactic structures," 1957.

[14] ——, "Aspects of the theory of syntax," 1965.

[15] ——, *The minimalist program*. MIT press, 1995.

[16] D. Harasim, C. Finkensiep, P. Ericson, T. J. O'donnell, and M. Rohrmeier, "The jazz harmony treebank," in *Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR*, 2020. [Online]. Available: https://github.com/DCMLab/JazzHarmonyTreebank

[17] J. Nivre, "Dependency grammar and dependency parsing," *MSI report*, vol. 5133, 2005.

[18] M. Steedman and J. Baldridge, *Combinatory Categorial Grammar*, 2011.

[19] Y. Ogura, H. Ohmura, Y. Uehara, H. Tojo, and K. Katsurada, "Expectation-based parsing for jazz chord sequences," in *Proceedings of the 17th Sound and Music Computing Conference*, 2020, pp. 350–356.

[20] M. Nagao *et al.*, *Natural Language Processing*. Iwanami Shoten co., 1996.

[21] D. Oyama, *Jazz Theory Workshop*. Musashino Conservatory Publishing Department, 2004.

[22] M. Levine, *The Jazz Theory Book*. Sher Music, 1995.

[23] E. Narmour, *The Analysis and Cognition of Basic Melodic Structures: The Implication-Realization Model*. The University of Chicago Press, 1992.

GENERALIZED TONAL PITCH SPACE WITH EMPIRICAL TRAINING

Hiroyuki YAMAMOTO (山本紘征) (yamamoto@kusuli.com) [1] and **Satoshi TOJO** (東条敏) (tojo@jaist.ac.jp) [1]

[1] **JAIST**, Ishikawa, Japan

ABSTRACT

A chord name can be interpreted in multiple ways, so a sequence of chord names has combinatorially many interpretations though most of which are inadequate. Tonal Pitch Space (TPS) is a music model which enables us to measure the distance between two chords, and thus we can rely on the theory to find most plausible interpretations, calculating the shortest path in the network of chord sequences. Although TPS is based on classical music theory, it is not based on data in a precise sense. As a result, the distance in the original TPS is somewhat rough to achieve high prediction accuracy.

In this study, we combine empirical observations with TPS, that is, to allow users to pick arbitrary combinations of features and calculate the distance of two chord interpretations. Then we propose a path probability formula to convert a path distance to a path probability, so that we can train the parameters from annotated datasets. We illustrate several experimental distance elements and show that some combinations of them can significantly improve the prediction accuracy, which resulted in over 86% in the test set.

1. INTRODUCTION

A Berklee style chord name by itself can be interpreted in several ways, and we need to consider the context to determine the plausibility of each candidate. Tonal Pitch Space (TPS) [3] gives us a foundation to consider the context by defining the smoothnesss of chord connection as the numeric distance between two chords, given their keys and degrees. Based on this, Sakamoto *et al.* [4] have proposed a method to find the most plausible interpretation path for a chord sequence as the shortest path in the interpretation graph, that expresses all possible chord interpretation paths each edge is weighted by the distance on TPS. However, the prediction accuracy of this method is only around 40%. This is, we assume, partly because TPS is based on classical music theory but not on data. So its structure and coefficients are not, strictly speaking, defined in an objective manner. Therefore, the model is a little too simple to achieve high prediction accuracy.

In this study, we work through these problems by combining empirical observations with TPS. First, we rearrange the distance formula in TPS to the sum of three distance elements, then generalize it to allow us to add other distance elements. These distance elements we define are in the form of tables whose cells correspond to the specific combinations of features of two chord interpretations. Next, we propose a path probability formula which gives higher probability to a path with shorter total distance. Finally, by differentiating the cross entropy loss function, we calculate the gradient and update the parameters using it.

Our approach [1] enables us to generalize and refine TPS by learning the metric model of arbitrary combinations of features as long as they contribute to decrease the value of target (loss) function. And we demonstrate the effectiveness of our approach by showing the best model being able to significantly improve the prediction accuracy and achieve over 86%.

This paper is organized as follows. In Section 2, we review related works. Then we give the formal representation of our proposed model and the learning strategy in Section 3 and 4, respectively. Thereafter, we show the experimental results in Section 5. Finally, we conclude in Section 6.

2. TPS-BASED APPROACH

There have been a lot of approaches to analyze musical harmony, and nowadays, a model with Hidden Markov Model (HMM) [12–15] and that with neural networks [16–18] seem prevalent. In this paper, however, we focus on Tonal Pitch Space. The theory finds the shortest path by the sum of the smallest distances in chords, and thus it results in the most plausible interpretation of chords by keys and degrees. Therefore, the detection of the shortest path is also expected to coincide with the local key identification.

2.1 Tonal Pitch Space

TPS is a music model for the quantitative harmony analysis proposed by Lerdahl [3]. It is proposed to complement Lerdahl's the other music theory (the Generative Theory of Tonal Music [5]), which applies the generative grammar to extend the Schenkerian theory. A chord can be interpreted in multiple degree/key pairs (*e.g.*, interpretations of C major triad are as follows: I/C, III/a, V/F, IV/G, VI/ a, and VII/d) and TPS defines a distance between every pair of these degree/key pairs.

The distance between chord interpretations x and y, when

[1] Source code is available at https://github.com/kusuli/smc2021/.

they are in related keys, can be calculated as equation (1)

$$\delta(x,y) = \begin{array}{r} region(x,y) + chord(x,y) \\ +basicspace(x,y) \end{array} \quad (1)$$

where $region(x,y)$ is a distance between keys, $chord(x,y)$ is a distance between degrees, and $basicspace(x,y)$ is a distance on a structure called basic space.

The calculation above is applicable only when x and y are in related keys which are defined as follows:

$$C(R) = \begin{cases} \{\mathrm{I}, \mathrm{i}, \mathrm{ii}, \mathrm{iii}, \mathrm{IV}, \mathrm{V}, \mathrm{vi}\} & \text{if } R \text{ is a major key} \\ \{\mathrm{i}, \mathrm{I}, \mathrm{bIII}, \mathrm{iv}, \mathrm{v}, \mathrm{bVI}, \mathrm{bVII}\} & \text{otherwise} \end{cases} \quad (2)$$

where roman numerals in this equation mean the keys with the tonic being the degree of R (*e.g.*, $C(\mathrm{F})$ is F, f, g, a, Bb, C, and d). If x and y are not in related keys (*i.e.*, distant keys), distance between x and y can be calculated as :

$$\delta(x,y) = \min\left(\begin{array}{l} \delta(x, T_{R_1}) + \Delta(R_1, R_n) + \delta(T_{R_n}, y) \\ |R_1 \in C(R_x), R_n \in C(R_y) \end{array} \right)$$

$$\Delta(R_1, R_n) = \min\left(\sum_{i=1}^{n-1} \delta(T_{R_i}, T_{R_{i+1}}) | R_{i+1} \in C(R_i)\right) \quad (3)$$

where T_R is key R's tonic, R_z is chord interpretation z's key. In other words, the transition from x to y must be considered as a combination of transitions within related keys, and the overall distance is the shortest total distance of the transitions.

As explained above, the distance within related keys (equation (1)) is composed of the sum of three elements. Now, because equation (3) is the sum of equation (1)s, the resulting distance can also be considered as the sum of three elements. Therefore, we can rewrite the distance as follows:

$$\delta(x,y) = \begin{array}{r} totalRegion(x,y) + totalChord(x,y) \\ +totalBasicspace(x,y) \end{array} \quad (4)$$

2.2 Former Approaches based on TPS

Sakamoto *et al.* [4] have applied TPS to analyze chord sequences to find the most plausible interpretation as the shortest path based on the distances described above.

Given a chord sequence, first their method extends each chord to its interpretations and constrcuts a graph whose edges have weights that correspond to the distances on TPS. Then it applies the Viterbi algorithm [6] to find the shortest interpretation paths from the start to the goal. Figure 1 shows an interpretation graph for chord sequence C → F → G → C. One of the shortest interpretation path in Figure 1 is I/C → IV/C → V/C → I/C.

Catteau *et al.* [9] utilized the key profiles of Temperly [10] alongside TPS to define probabilities concerning chords, scales, and chroma vectors to estimate keys and chords from audio. Rocher *et al.* [11] used Temperly's key profiles and TPS to construct a harmonic graph then estimate individual chords and keys by finding the best path.

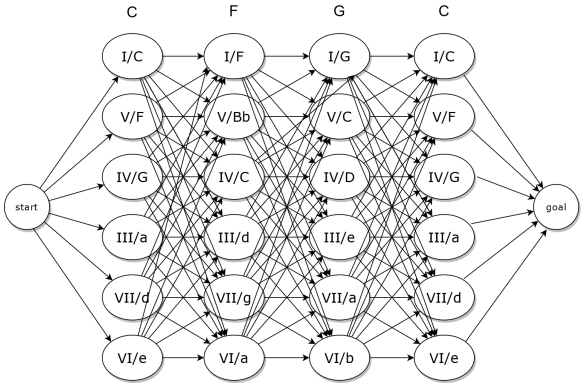


Figure 1. Interpretation graph.

In the effort to improve cadence detection, Matsubara *et al.* [7] have proposed to restrict the minor scale to harmonic one to avoid the ambiguity in chord interpretation, and to revise the candidates of chord interpretations of each chord names. Yamamoto *et al.* [8] have proposed to extend TPS and interpretation graph to consider (1) tetrads and three minor scales, (2) pivot-chord modulations, and (3) certain cadence patterns to improve the expressiveness and reduce the ambiguity mainly focusing on jazz harmony. Furthermore, there are many approaches with some kinds of metric models other than TPS. Feisthauer *et al.* [19] ,for example, defined three proximity measures based on musicologial knowledge to find the optimal path as the tonal plan.

In the following sections, we revise the structure of TPS and predict chord interpretations using the interpretation graph proposed by Sakamoto *et al.* [4].

3. PROPOSED MODEL

We define notations as follows:

$\mathcal{X} \triangleq \{\mathrm{I/A}, \mathrm{ii/A}, \cdots, \mathrm{VI/g\#}, \mathrm{VII/g\#}\}$: the set of chord interpretations

$x, y \in \mathcal{X}$: individual chord interpretations

$\mathcal{I} \triangleq \{1, 2, 3, 4.1, 4.2, 5.1, \cdots, |\mathcal{I}|\}$: the set of distance element indices

$scale : \mathcal{X} \to \{0, 1\}$: the function which maps a chord interpretation to its scale [2] (*e.g.* $scale(iii/A) = 0, scale(III/c) = 1$)

$tonic : \mathcal{X} \to \{n \in \mathbb{Z} | 0 \leq n \leq 11\}$: the function which maps a chord interpretation to its tonic note [3] (*e.g.* $tonic(iii/A) = 9, tonic(III/c) = 0$)

$$majorTonic(x) \triangleq \begin{cases} tonic(x) \text{ if } scale(x) = 0 \\ (tonic(x) + 3)\ mod\ 12 \text{ otherwise} \end{cases}$$

[2] Here, we only consider major (= 0) and minor (= 1) scales.
[3] We use pitch classes to express notes.

(*e.g.* $majorTonic(iii/A) = 9$, $majorTonic(III/c) = 3$)

$root : \mathcal{X} \to \{n \in \mathbb{Z} | 0 \leq n \leq 11\}$: the function which maps a chord interpretation to its root note (*e.g.* $root(iii/A) = 1$, $root(III/c) = 3$)

$degree : \mathcal{X} \to \{n \in \mathbb{Z} | 1 \leq n \leq 7\}$: the function which maps a chord interpretation to its degree (*e.g.* $degree(iii/A) = 3$, $degree(III/c) = 3$)

$distanceElement_i : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$: the function which maps a chord interpretation pair to their distance based on the distance element of index $i \in \mathcal{I}$

$b : \mathcal{I} \to \{0, 1\}$: the function which specifies the activation of each distance element

The distance on TPS can be thought of the sum of three distance elements as in equation (4). Now we rearrange this equation as a sum of all (active) distance elements.

$$GTPS(x, y) = \sum_{i \in \mathcal{I}} b(i) \cdot distanceElement_i(x, y) \quad (5)$$

The first three distance elements are from the original TPS, namely, $totalRegion$, $totalChord$, and $totalBasicspace$ in equation (4). In addition to them, we can add arbitrary new distance elements by freely choosing which and which features to distinguish. In the following subsections, we propose in total twelve new distance elements, which are inspired by the original TPS. Finally, with $b(i)$ term in equation (5), we can use any combinations of distance elements.

3.1 Distance Element 4: Scale Distance

Distance elements for scale transitions. We define two variants as follows:

3.1.1 DE 4.1: Symmetric Scale Distance

$$M_{4.1} \in \mathbb{R}^2$$
$$distanceElement_{4.1}(x, y) \triangleq M_{4.1} \begin{bmatrix} (scale(x) - scale(y)) \\ mod\ 2 \end{bmatrix}^{4} \quad (6)$$

This one merely distinguishes whether the scale is changed or not.

3.1.2 DE 4.2: Asymmetric Scale Distance

$$M_{4.2} \in \mathbb{R}^{2 \times 2}$$
$$distanceElement_{4.2}(x, y) \triangleq M_{4.2} [scale(x), scale(y)] \quad (7)$$

The asymmetric version of DE 4.1 (*e.g.*, major → minor, and minor → major are considered same in DE 4.1, but not in DE 4.2).

[4] $M[i_i, i_2, \cdots, i_n]$ indicates the value in n dimensional table M at the index $(i_1, i_2, \cdots, i_n)$.

3.2 Distance Element 5: Tonic Distance

Distance elements for tonic transitions, by which we intend to generalize $totalRegion$ in equation (4). Tonic transitions can be thought of as key transitions without considering scales. We define six variants as follows:

3.2.1 DE 5.1: Symmetric Relative Tonic Distance

$$M_{5.1} \in \mathbb{R}^7$$
$$distanceElement_{5.1}(x, y) \triangleq M_{5.1} \left[min \begin{pmatrix} (majorTonic(y) - majorTonic(x)) \\ mod\ 12, \\ (majorTonic(x) - majorTonic(y)) \\ mod\ 12 \end{pmatrix} \right] \quad (8)$$

Among all variants, this one is conceptually closest to the original $totalRegion$.

3.2.2 DE 5.2: Symmetric Parallel Tonic Distance

$$M_{5.2} \in \mathbb{R}^7$$
$$distanceElement_{5.2}(x, y) \triangleq M_{5.2} \left[min \begin{pmatrix} (tonic(y) - tonic(x))\ mod\ 12, \\ (tonic(x) - tonic(y))\ mod\ 12 \end{pmatrix} \right] \quad (9)$$

Unlike the relative tonic distance, this one identifies parallel keys (*e.g.*, C major and C minor), instead of relative keys (*e.g.*, C major and A minor).

3.2.3 DE 5.3: Asymmetric Relative Tonic Distance

$$M_{5.3} \in \mathbb{R}^{12}$$
$$distanceElement_{5.3}(x, y) \triangleq M_{5.3} [(majorTonic(y) - majorTonic(x))\ mod\ 12] \quad (10)$$

The asymmetric version of DE 5.1 (*e.g.*, C → D and D → C are distinguished in DE 5.3, but not in DE 5.1).

3.2.4 DE 5.4: Asymmetric Parallel Tonic Distance

$$M_{5.4} \in \mathbb{R}^{12}$$
$$distanceElement_{5.4}(x, y) \triangleq M_{5.4} [(tonic(y) - tonic(x))\ mod\ 12] \quad (11)$$

The asymmetric version of DE 5.2.

3.3 Distance Element 6: Key Distance

Distance elements for key transitions, which can handle both scale transitions and tonic transitions at once. One can calculate those distances by the combination of DE 4.x and DE 5.x, but this assumes the independence of the transitions of scales and that of tonics. By contrast, DE 6.x can consider the interactions of scales and tonics, if any. There are two variants as follows:

3.3.1 DE 6.1: Symmetric Key Distance

$$M_{6.1} \in \mathbb{R}^{2\times 7}$$
$$distanceElement_{6.1}(x, y)$$
$$\triangleq M_{6.1} \begin{bmatrix} (scale(x) - scale(y)) \\ \quad mod\ 2, \end{bmatrix} min \begin{pmatrix} (tonic(y) - tonic(x)) \\ \quad mod\ 12, \\ (tonic(x) - tonic(y)) \\ \quad mod\ 12 \end{pmatrix} \tag{12}$$

3.3.2 DE 6.2: Asymmetric Key Distance

$$M_{6.2} \in \mathbb{R}^{2\times 2\times 12}$$
$$distanceElement_{6.2}(x, y)$$
$$\triangleq M_{6.2} \begin{bmatrix} scale(x), scale(y), \\ \end{bmatrix} \begin{matrix} (tonic(y) - tonic(x)) \\ mod\ 12 \end{matrix} \tag{13}$$

The asymmetric version of DE 6.1.

3.4 Distance Element 7: Root-Degree Distance

Distance elements for root note transitions from each degree, which roughly generalize $totalChord$ in equation (4), although with much more information [5]. We define two variants as follows:

3.4.1 DE 7.1: Symmetric Root-Degree Distance

$$M_{7.1} \in \mathbb{R}^{7\times 7}$$
$$distanceElement_{7.1}(x, y)$$
$$\triangleq M_{7.1} \begin{bmatrix} degree(x), min \begin{pmatrix} (root(y) - tonic(x)) \\ \quad mod\ 12, \\ (tonic(x) - root(y)) \\ \quad mod\ 12 \end{pmatrix} \end{bmatrix} \tag{14}$$

This one calculates distance according to the relative positions of roots for each (source) degree.

3.4.2 DE 7.2: Asymmetric Root-Degree Distance

$$M_{7.2} \in \mathbb{R}^{7\times 12}$$
$$distanceElement_{7.2}(x, y)$$
$$\triangleq M_{7.2} \left[degree(x), (root(y) - tonic(x))\ mod\ 12\right] \tag{15}$$

The asymmetric version of DE 7.1.

3.5 Distance Element 8: Key-Degree Distance

Distance elements for key and degree transitions, which can handle both key (*i.e.*, scale and tonic) transitions and degree transitions all at once. Unlike the combinations of DE 4.x, DE 5.x, and DE 7.x or DE 6.x and DE 7.x, DE 8.x can consider the interactions of scales, tonics and degrees. We define two variants as follows:

[5] A straightforward way to do this may be to take step distance between two degrees (*i.e.*, replacing $tonic$ in DE 5.2 and DE 5.4 with $degree$), but we omitted them because both of them perform very poorly.

3.5.1 DE 8.1: Symmetric Key-Degree Distance

$$M_{8.1} \in \mathbb{R}^{2\times 7\times 7\times 7}$$
$$distanceElement_{8.1}(x, y)$$
$$\triangleq M_{8.1} \left[\ (scale(x) - scale(y))\ mod\ 2, degree(x),\right.$$
$$\left. degree(y), min \begin{pmatrix} (tonic(y) - tonic(x))\ mod\ 12, \\ (tonic(x) - tonic(y))\ mod\ 12 \end{pmatrix} \right] \tag{16}$$

3.5.2 DE 8.2: Asymmetric Key-Degree Distance

$$M_{8.2} \in \mathbb{R}^{2\times 7\times 2\times 12\times 7}$$
$$distanceElement_{8.2}(x, y)$$
$$\triangleq M_{8.2} \left[scale(x), degree(x), scale(y), degree(y),\right.$$
$$\left. (tonic(y) - tonic(x))\ mod\ 12\right] \tag{17}$$

The asymmetric version of DE 8.1.

All the proposed distance elements and sample indices are listed in Table 1.

DE	I/C → V/G	I/C → iv/c#
4.1 Sym Scale	(0)	(1)
4.2 Asym Scale	(0, 0)	(0, 1)
5.1 Sym Relative Tonic	(5)	(4)
5.2 Sym Parallel Tonic	(5)	(1)
5.3 Asym Relative Tonic	(7)	(4)
5.4 Asym Parallel Tonic	(7)	(1)
6.1 Sym Key	(0, 5)	(1, 1)
6.2 Asym Key	(0, 0, 7)	(0, 1, 1)
7.1 Sym Root-Degree	(1, 2)	(1, 6)
7.2 Asym Root-Degree	(1, 2)	(1, 6)
8.1 Sym Key-Degree	(0, 1, 5, 5)	(1, 1, 4, 1)
8.2 Asym Key-Degree	(0, 1, 0, 5, 7)	(0, 1, 1, 4, 1)

Table 1. Indices for two sample transitions.

4. LEARNING STRATEGY

We define additional notations as follows:

G: an interpretation graph with $|G|$ layers

$G_{s:t}$: from sth layer to tth layer of G ($G_{s:s}$ can be abbreviated as G_s). As a simplified notation, a node in the sth layer can be written as $x \in G_s$, likewise, $x \in G_{s:t}$ be a path from the sth layer to the tth layer, and $x \in G_{s:t-1}||x_t$ be a path from sth layer to the $(t-1)$th layer and added x_t to be the last node.

$x_{s:t}$: from sth element to tth element of an interpretation path $x_{0:|G|}$ ($x_{s:s}$ can be abbreviated as x_s)

$x^*_{0:|G|}$: the ground truth interpretation path

$GTPS_{path}(x_{s:t}) \triangleq \sum_{u=s}^{t-1} GTPS(x_u, x_{u+1})$

We want the calculated distances to allow us to estimate the true interpretation path as a shortest path in the interpretation graph. So we need to learn the parameters to give true interpretation path a shorter total distance than the other interpretation paths.

For that purpose, we first define the path probability formula and then train the tables by using the gradients on the parameter spaces.

4.1 Path Probability

We define the path probability from start node to sth chord interpretation as below:

$$\begin{aligned}&\mathrm{P}(X_{0:s} = x_{0:s}|G_{0:s})\\ &\triangleq \begin{cases} 1 & \text{if } s = 0\,^{6} \\ \prod_{t=0}^{s-1} \frac{\exp(-GTPS(x_t, x_{t+1}))}{Z_{G,t}} & \text{otherwise} \end{cases}\end{aligned} \tag{18}$$

where

$$Z_{G,t} \triangleq \sum_{l \in G_t} \sum_{m \in G_{t+1}} \mathrm{P}(X_t = l|G_{0:t}) \exp(-GTPS(l, m))$$

We can calculate the probability for the whole interpretation path as $\mathrm{P}(X_{0:|G|} = x_{0:|G|}|G_{0:|G|})$. This probability is designed to give higher values to the interpretation paths with shorter total distances (Theorem 1).

We can calculate the node probability $\mathrm{P}(X_s = x_s|G_{0:s})$ by marginalizing path probability:

$$\begin{aligned}\mathrm{P}(X_s = x_s|G_{0:s}) &= \sum_{x_{0:s} \in G_{0:s-1}||x_s} \mathrm{P}(X_{0:s} = x_{0:s}|G_{0:s}) \\ &= \sum_{x_{0:s} \in G_{0:s-1}||x_s} \prod_{t=0}^{s-1} \frac{\exp(-GTPS(x_t, x_{t+1}))}{Z_{G,t}} \\ &= \sum_{x_{0:s} \in G_{0:s-1}||x_s} \left(\prod_{t=0}^{s-2} \frac{\exp(-GTPS(x_t, x_{t+1}))}{Z_{G,t}} \right) \\ &\quad \times \frac{\exp(-GTPS(x_{s-1}, x_s))}{Z_{G,s-1}} \\ &= \sum_{x_{0:s} \in G_{0:s-1}||x_s} \mathrm{P}(X_{0:s-1} = x_{0:s-1}|G_{0:s-1}) \\ &\quad \times \frac{\exp(-GTPS(x_{s-1}, x_s))}{Z_{G,s-1}} \\ &= \sum_{x_{s-1} \in G_{s-1}} \mathrm{P}(X_{s-1} = x_{s-1}|G_{0:s-1}) \\ &\quad \times \frac{\exp(-GTPS(x_{s-1}, x_s))}{Z_{G,s-1}}\end{aligned}$$

When $s = 0$, $x_{0:0} \in G_{0:-1}||x_0$ becomes $x_{0:0} \in x_0$ because $G_{0:-1}$ is empty. Note that, $\mathrm{P}(X_s = x_s|G_{0:s}) = \mathrm{P}(X_s = x_s|G_{0:|G|})$ is not always the case. As we can see, this process has a recursive structure, and, by calculating and memorizing in a sequential manner from the start node, we can get the node probability and path probability with the time complexity linear to s.

[6] 0th layer contains only one node, that is, the start node

4.2 Loss and Gradient

We define a cross entropy loss function as follows:

$$\begin{aligned}&Loss(x_{0:|G|}|G_{0:|G|}) \\ &\triangleq \sum_{x_{0:|G|} \in G_{0:|G|}} \begin{matrix} -\mathrm{P}^*(X_{0:|G|} = x_{0:|G|}) \\ \times \ln \mathrm{P}(X_{0:|G|} = x_{0:|G|}|G_{0:|G|}) \end{matrix}\end{aligned} \tag{19}$$

Here, P^* is the probability function which only responds to the ground truth:

$$\mathrm{P}^*(X_{0:|G|} = x_{0:|G|}) \triangleq \begin{cases} 1 \text{ if } x_{0:|G|} = x^*_{0:|G|} \\ 0 \text{ otherwise} \end{cases}$$

We can get the gradient by differentiating $Loss$ (19) with respect to the parameters, then apply stochastic gradient descent algorithm to update the parameters to minimize the value of $Loss$ (19), which results in maximizing the path probability for the ground truth path.

4.3 Accuracy

We evaluate our model based on how accurately it can predict each chord interpretation by specifying the shortest path in the interpretation graph. If there are more than one shortest paths, we calculate a weighted average for each node in proportion [7] to how many paths go through the node as is illustrated in Figure 2.

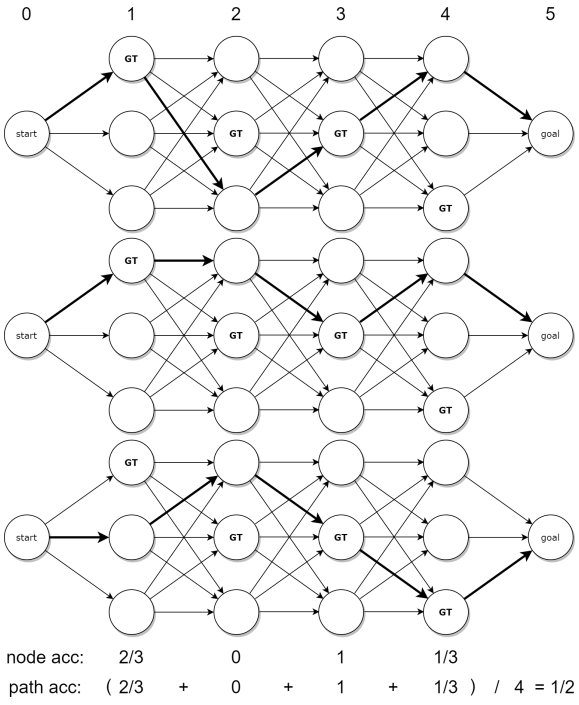


Figure 2. Accuracy calculation when there are more than one shortest paths.

[7] this proportion is different from the node probability

5. EXPERIMENTS

5.1 Data and Method

We use the dataset annotated in *rntxt* format [1], published at [1, 2]. The dataset is composed of 360 pieces (1,691 phrases, 76,341 chords) and we regard every phrase as a unit (*i.e.*, to which we predict the interpretation path) but when a phrase exceeds 50 chords we divide it into units each of which does not exceed 50 chords, resulting in 2,472 phrases. Then use 1,976 phrases to the training, and 248 phrases to the validation, and remaining 248 phrases to the test

Rntxt format contains a lot of information other than degree/key, but in this study we utilize only key and degree information. About secondary/tertiary chords, we employ local keys (*e.g.*, V/V/V on C major key is interpreted as V on D major key).

We set all initial parameter values to be zero and train the models by mini-batch stochastic gradient descent with batch size=100 and learning rate=0.001. We continue training until no accuracy update in validation set for ten epochs in a row, then pick the parameter which gives the highest validation accuracy..

5.2 Results

We compare the performances of each distance element and some combinations. The result is shown in the Table 2, 3, and 4.

ex	DE 1	DE 2	DE 3	mean	stdev
0				0.1900	0.0257
1	○	○	○	0.3847	0.1023
2	○			0.3780	0.1034
3		○		0.1930	0.0288
4			○	0.3842	0.1023
5	○	○		0.3770	0.1052
6	○		○	0.3850	0.1025
7		○	○	0.3841	0.1023

Table 2. Performances of each distance element (and combinations) of original TPS. [9]

ex 0 is without any distance elements, just for information.

ex 1 is the original TPS. This one successfully double the accuracy (*i.e.*, narrow down the candidate interpretation by half) from **ex 0**. We consider this one to be the baseline.

We also conduct ablation patterns of TPS (**ex 2-7**). When used alone (**ex 2-4**), $totalBasicspace$ is the best performance (**ex4**) and achieved almost the same accuracy as the full TPS (**ex 1**). We consider the reason why $totalBasicspace$ is a little better than $totalRegion$ (**ex 2**) is that basic space can exress region distance by the diatonic level and also other levels can give additional information. Seeing the result of **ex 3**, however, $totalChord$ do not improve accuracy well. That is also the case when used two of them together (**ex 5-7**).

[9] **ex**, **DE**, **mean**, and **stdev** represent experiment, distance element, model mean accuracies, and standard deviations of accuracies respectively

ex	DE	prms	mean	stdev
8	4.1 Sym Scale	2	0.1900	0.0257
9	4.2 Asym Scale	4	0.2522	0.1432
10	5.1 Sym Relative Tonic	7	0.3983	0.1006
11	5.2 Sym Parallel Tonic	7	0.2908	0.2415
12	5.3 Asym Relative Tonic	12	0.3974	0.1003
13	5.4 Asym Parallel Tonic	12	0.2870	0.2408
14	6.1 Sym Key	14	0.4249	0.1739
15	6.2 Asym Key	48	0.5017	0.3380
16	7.1 Sym Root-Degree	49	0.5646	0.1640
17	7.2 Asym Root-Degree	84	0.5741	0.1628
18	8.1 Sym Key-Degree	686	0.8625	0.1780
19	8.2 Asym Key-Degree	2,352	0.8690	0.1717

Table 3. Performances of proposed distance elements.

In **ex 8-19**, we test each proposed distance elements by themselves. DE 5.1 can accomplish almost the same accuracy as the full TPS (**ex 1, 10**), although it has only seven parameters. DE 4.x cannot improve accuracy at all without distinguishing directions (**ex 8,9**), but surprisingly, for many other distance elements, it turns out that there is very little or no accuracy gain by distinguishing the direction from the comparisons **ex 10** to **ex 12**, **ex 11** to **ex 13**, **ex 16** to **ex 17**, and **ex 18** to **ex 19**. We also test tonic distances in which parallel keys are identified (**ex 11, 13**), but they are significantly worse than those of relatie keys (**ex 10, 12**). DE 8.x, being the most complex distance elements, can achieve over 86% accuracy.

In **ex 20-26**, we test some combinations of proposed distance elements. The combinations are selected so that involved distance elements complement each other though not exhaustive. The combination of **ex 23** can achieve 83% with only 58 parameters, likewise, that of **ex 25** and **ex 26** can achieve 85.0% and 86% with a little more parameters. Therefore, it seems that taking the interactions of all scale, tonic, and degree into account is not so important considering the huge parameter size. Also, it is interesting that DE 4.1 have meaningful contribution in **ex 23** here although it does not make difference at all by itself (**ex 8**).

We also test some combinations of TPS element and distance tables (**ex 27-29**). Root table can be benefited from the elements from TPS (**ex 28**), but in the other combinations, there are not so obvious accuracy gains.

From all the experiments, we can observe that the combinations which achieved over 80% (**ex 18, 19, 23-26, 29**) have all three features (*i.e.*, $scale$, $tonic/majorTonic$, and $degree$ (or $degree, root$)), but one of them (**ex 23**) does not consider interactions nor directions. Therefore, it seems that, including those three features is crucial, but considering interactions or directions have relatively small effects.

For illustrative purpose, we show some possible interpretations for a chord progression Cm $\rightarrow$ F $\rightarrow$ Bb $\rightarrow$ Eb $\rightarrow$ A$^\circ$ $\rightarrow$ D $\rightarrow$ Gm and their total distances in Table 5. We

ex	DE 1	DE 2	DE 3	DE 4.1	DE 4.2	DE 5.1	DE 6.1	DE 6.2	DE 7.1	DE 7.2	prms	mean	stdev
20				○		○					9	0.3978	0.1015
21					○	○					11	0.5131	0.3402
22						○			○		56	0.7627	0.1585
23				○		○			○		58	0.8301	0.1869
24					○	○			○		60	0.8226	0.1814
25							○		○		63	0.8495	0.1775
26								○		○	132	0.8601	0.1681
27		○	○				○				14	0.4210	0.2318
28	○		○						○		49	0.7309	0.1566
29			○				○		○		63	0.8308	0.1887

Table 4. Performances of some combinations of distance elements.

use the model trained in **ex 23** to calculate the distances. In this example, paths **b** and **c** both have only one key, but calculated distances are longer than that of **a**, which consists of two keys. But they are shorter than paths **d** and **e**. We think this order more or less matches to our musical perception.

	Cm	F	Bb	Eb	A°	D	Gm
a	ii/Bb	V/Bb	I/Bb	VI/g	ii°/g	G/g	i/g
	-	5.88	8.91	16.44	22.30	28.11	31.14
b	ii/Bb	V/Bb	I/Bb	IV/Bb	vii°/Bb	III/Bb	vi/Bb
	-	5.88	8.91	14.25	20.75	26.59	33.36
c	iv/g	VII/g	III/g	VI/g	ii°/g	G/g	i/g
	-	6.27	12.66	19.64	25.50	31.31	34.34
d	v/f	I/F	IV/F	VII/f	vii°/Bb	VI/F	ii/F
	-	8.77	14.11	26.12	37.46	46.03	51.89
e	i/c	I/F	I/Bb	I/Eb	ii°/g	I/D	i/g
	-	10.84	18.91	26.98	37.41	49.12	60.29

Table 5. Some possible interpretations and their total distances.

6. CONCLUSIONS

In this study, we have extended TPS to take in empirical observation. We generalized the distance formula in TPS and proposed a way to define distance elements that distinguish any combinations of given features and to train them with data. Our best combination achieved 86.9% accuracy in the test set, which is significantly higher than that of the baseline model (38.5%), and this result, we believe, shows that our approach successfully learns an effective metric structure from data. Also, one of our combination with only 58 parameters achieved 83%, and with 132 parameters, 86%. We hope that these simple models will help us to understand better about the structure of tonal harmony.

There are many potential directions to improve our method. First, it would be beneficial to accept sequences of chroma vectors or piano-roll as input. Second, not only TPS, it would also be meaningful to extend our approach to deal with key profiles like Krumhansl's [20]. Furthermore, the fact that distinguishing directions only makes small difference in accuracy is somewhat contradictory to our previous research [8]. This implies that there may be a better way to take directions into account.

Acknowledgments

This work is supported by JSPS Kakenhi 16H01744.

7. REFERENCES

[1] D. Tymoczko, M. Gotham, M. S. Cuthbert, and C. Ariza. "The RomanText Format: A Flexible and Standard Method for Representing Roman Numeral Analyses". *International Society for Music Information Retrieval Conference (ISMIR)*, pp.123-129, 2019

[2] M. S. Cuthbert, C. Ariza. "music21: A toolkit for computer-aided musicology and symbolic music data.", *International Society for Music Information Retrieval Conference (ISMIR)*, pp.637–642, 2010

[3] F. Lerdahl: "Tonal Pitch Space", Oxford University Press, 2001

[4] S. Sakamoto, S. Arn, M. Matsubara, S. Tojo: "Harmonic analysis based on tonal pitch space", in *Proceedings of the 8th International Conference on Knowledge and Systems Engineering (KSE)*, pp.230-233, 2016

[5] F. Lerdahl, R. Jackendoff: *"A Generative Theory of tonal music"*, Cambridge, MA, 1983

[6] A. Viterbi: *"Error bounds for convolutional codes and an asymptotically optimum decoding algorithm." IEEE transactions on Information Theory*, 13.2: pp.260-269, 1967

[7] M. Matsubara, T. Kodama, S. Tojo: *"Revisiting cadential retention in GTTM"* in *2016 Eighth international conference on knowledge and systems engineering (KSE)*, pp.218-223, 2016

[8] H. Yamamoto, Y. Uehara, S. Tojo: *"Jazz harmony analysis with ϵ-transition and cadential shortcut"* in *Proceedings of 17th Sound and Music Computing Conference (SMC)*, pp.316-322, 2020

[9] B. Catteau, J. Martens, M. Leman. *"A probabilistic framework for audio-based tonal key and chord recognition"* in *Advancesin Data Analysis*, pp.637–644, 2007

[10] D Temperley. *"What's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered"*. Music Perception 17, 1, pp.65–100. 1999

[11] T. Rocher, M. Robine, P. Hanna, L. Oudre. *"Concurrent estimation of chords and keys from audio"*. in *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, pp.141–146. 2010

[12] W. Chai, B. Vercoe. *"Detection of key change in clssical piano music"*. in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pp.468–473. 2005

[13] L. Mearns, E. Benetos, S. Dixon. *"Automatically detecting key modulations in J. S. Bach Chorale recordings"*. in *Proceedings ofthe 8th Sound and Music Computing Conference (SMC)*, pp.25–32, 2011

[14] N. Nápoles López, C. Arthur, I. Fujinaga. *"Key-finding based on a hidden Markov model and key profiles"*. in *Proceedings of the 6th International Conference on Digital Libraries for Musicology*, pp.33–37, 2019

[15] H. Papadopoulos, G. Peeters. *"Local Key Estimation Based on Harmonic and Metric Structures"*. in *International Conference on Digital Audio Effects (DAFx)*, pp. 408–415, 2009

[16] T. Chen, L Su. *"Functional harmony recognition of symbolic music data with multi-task recurrent neural networks"*. in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, pp.90–97, 2018

[17] T. Chen, L. Su. *"Harmony transformer: incorporating chord segmentation into harmony recognition"*. in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, pp.259–267, 2019

[18] G. Micchi, M. Gotham, M Giraud. *"Not all roads leadto Rome: pitch representation and model architecture for automatic harmonic analysis*, in *Transactions of the International Society for Music Information Retrieval* 3, 1 (May 2020), pp.42–54, 2020

[19] L. Feisthauer, L. Bigo, M. Giraud, F. Levé. *"Estimating keys and modulations in musical pieces"*. in *Sound and Music Computing Conference (SMC)*, pp. 323-330, 2020

[20] C. L. Krumhansl *"Cognitive foundations of musical pitch"*, Oxford University Press, 1990

8. APPENDIX

Theorem 1 (order accordance)**.** *In an interpretation graph G, $GTPS_{path}(x_{0:s})$ is smaller than $GTPS_{path}(x'_{0:s})$ if and only if $\mathrm{P}(X_{0:s} = x_{0:s}|G_{0:s})$ is greater than $\mathrm{P}(X_{0:s} = x'_{0:s}|G_{0:s})$.*

Proof.

$$
\begin{aligned}
&GTPS_{path}(x_{0:s}) < GTPS_{path}(x'_{0:s}) \\
&\Leftrightarrow \exp(-GTPS_{path}(x_{0:s})) > \exp(-GTPS_{path}(x'_{0:s})) \\
&\Leftrightarrow \exp(-\sum_{t=0}^{s-1} GTPS(x_t, x_{t+1})) \\
&\quad > \exp(-\sum_{t=0}^{s-1} GTPS(x'_t, x'_{t+1})) \\
&\Leftrightarrow \prod_{t=0}^{s-1} \exp(-GTPS(x_t, x_{t+1})) \\
&\quad > \prod_{t=0}^{s-1} \exp(-GTPS(x'_t, x'_{t+1})) \\
&\#\text{divide both sides by the same (positive) value} \\
&\Leftrightarrow \frac{\prod_{t=0}^{s-1} \exp(-GTPS(x_t, x_{t+1}))}{\prod_{t=0}^{s-1} Z_{G,t}} \\
&\quad > \frac{\prod_{t=0}^{s-1} \exp(-GTPS(x'_t, x'_{t+1}))}{\prod_{t=0}^{s-1} Z_{G,t}} \\
&\Leftrightarrow \prod_{t=0}^{s-1} \frac{\exp(-GTPS(x_t, x_{t+1}))}{Z_{G,t}} \\
&\quad > \prod_{t=0}^{s-1} \frac{\exp(-GTPS(x'_t, x'_{t+1}))}{Z_{G,t}} \\
&\#\text{from equation (18)} \\
&\Leftrightarrow \mathrm{P}(X_{0:s} = x_{0:s}|G_{0:s}) > \mathrm{P}(X_{0:s} = x'_{0:s}|G_{0:s})
\end{aligned}
$$

□

THE UNREAL BOOK. ALGORITHMIC COMPOSITION FOR JAZZ LEAD SHEETS

Andrea VALLE (andrea.valle@unito.it) [1]

[1] *CIRMA/StudiUm*, **Università di Torino**, Italy

ABSTRACT

This paper presents the Unreal Book project which aims at exploring algorithmic generation of jazz lead sheets. Lead sheets are the standard notation format in jazz composition and are collected in many publications, the most relevant being the Real Book. Lead sheet format provides simple constraints (melody and chords) that allow for the application of algorithmic composition techniques based on the formalization of various jazz concepts. A computer-aided solution for the generation of a Real Book-like collection of lead sheets is presented, that takes also into account notation, including visual features that are considered defining of the Real Book. Seven examples of composition applications are shown, ranging from the implementation of jazz-inspired techniques to corpus-driven procedures.

1. BETWEEN COMPOSITION AND IMPROVISATION: LEAD SHEETS

Jazz has often been, and still is, largely described as an oral/aural practice, in which direct and mediated listening plays a pivotal role [1, 2]. This evidently holds true if one considers, respectively, the importance of musicians jamming together and of learning by imitation, and the relevance of recordings in the worldwide diffusion of jazz. Nevertheless, it has been observed that such an emphasis on oral/aural tradition has overshadowed the fundamental role that written sources in music notation have played since the early age of jazz, both in terms of organization of music structures (and thus performances) and diffusion of jazz repertoires [3]. The main written sources in jazz practice are lead sheets [4, 5]. Lead sheets are based on a notation format originating from classic American song. As they come from singing, they include three elements. The first two are the vocal melody, typically notated in the treble key, and the lyrics to be sung; the third is the harmonic background. Harmonies, i.e. chords, have been notated by means of a specific notation format, partly inherited from baroque abbreviated notation for basso continuo, but modified to take into account post-impressionistic harmonies [6], as shown in Figure 1. Chord sequences are known as "changes" in jazz practice. As jazz is mostly instrumental, most of the times lyrics have been dropped.

In short, the lead sheet notation format is a bare-bone one, including a single melody notated in common practice notion, with alphanumeric symbols on top representing chords. Apart from title and author, other typical features are an approximate indication tempo, and form abbreviations (da capo, section labels). An example is shown in Figure 2. Lead sheets are at the core of jazz practice as they mediate between composition and improvisation. On one side, they are written sources that provide input information to be taken into account by performers, like in classical Western composition. At the same time, many lead sheets are transcribed from recordings, so they do not share the same status of classical composed pieces, as they are rather scored a posteriori, even if the title and the author are referred to. On the other side, they are taken into account by performers as a starting material that is reorganized in various ways. Melodies can be modified by changing key, pitches and rhythm. Not only harmonies can be transposed to a new key, but they can also be radically altered, a standard practice known as "chord substitution" (see e.g. [8–10]). Form is only hinted at by lead sheets, typically showing the 32-bar form of the classic American song or the 12-bar form of blues: but these structures can be seen as starting points to be extended by intros, outros, soloing blocks, variable repetitions and reorganizations. Finally, lead sheets do not include arrangement features (orchestration), that are to be decided by the performer/arranger. To sum up, lead sheets, while still residing on the composition side, thanks to the openness of the format, propel a whole set of activities, leading to the final performance and steering to the improvisation side. This set of activities, placed in a crucial grey zone between composition and performance/improvisation, has been called "precomposition" [11]. Differently from written scores in Western classical music, lead sheets provide features and constraints that prompt, but not entirely determine, the final performance.

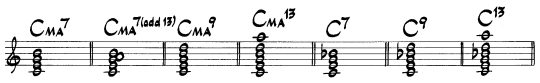


Figure 1. Some chords abbreviations and pitch content [7].

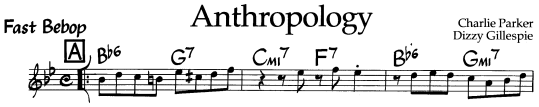


Figure 2. Beginning of *Anthropology*, from [7].

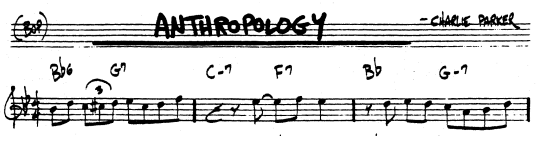


Figure 3. Beginning of *Anthropology*, from the *Real Book*.

2. FAKE, REAL, NEW REAL BOOKS

Lead sheets from famous jazz compositions have been shared in the jazz community, and soon they have been grouped into printed collections, reflecting the relevance in jazz practice of the repertoire of the so-called "standards" [12]. Printed collections of standards are typically named "fake books", a fake book being "a collection of charts or lead sheets used by jazz musicians (so-called because jazz musicians improvise, or "fake," their way through a performance)" [2, Appendix, p. 15]. The most famous fake book is the (pun intended) *Real Book*. The history of the latter is mostly unknown and goes unaddressed in all the recent histories of jazz. Apart some older pieces, it contains a selection of bebop and post-bop tunes, biased towards the late '60s-early '70s. The Real Book is a pirate assemblage of almost 5-hundreds, ad hoc handwritten transcriptions (including some well-known mistakes). Despite its obscure origins (it does not report any editorial data), the Real Book has gained a worldwide diffusion in the jazz community, first by means of photocopies, then in PDF format shared through p2p networks, thus becoming a de facto standard. The model is now at the core of jazz practice, as witnessed by the flowering of legal collections inspired by it (e.g. [7], Figure 2), including the authorized version by Hal Leonard of the very Real Book, providing the same cover, song list and typeface [13]. The Real Book handwritten typeface, while rooted into popular music arrangements, has become so iconic that it has been associated to the specific jazz flavour of music fonts as currently available in music notation software packages. Figure 2 is evidently inspired by Figure 3 from the original Real Book. The main features of the Real book are:

- dimension in order of 5-hundred of pieces;
- alphabetical indexing of the pieces;
- heterogeneous, even if not representative [3], sampling of jazz repertoire in terms of history, style, composition techniques;
- homogeneous notation format, based on lead sheet notation;
- hanwdritten "jazz" typeface.

3. THE UNREAL BOOK PROJECT

The Unreal Book Project is an algorithmic music composition project inspired by the Real Book. It aims at generating jazz-inspired compositions, notated in the lead sheet format, collected into a coordinated volume. The project focuses on three main objectives:

1. **Music composition**: as far as the author knows, no other project has focused on exploring the features and constraints of Real Book format in the context of algorithmic symbolic composition (i.e. resulting into music notation generation, see [14] rather than [15]). Algorithmic approaches to symbolic composition, while having a long and flourishing history, have never been applied to lead sheet generation [16, 17]. Of course, jazz has been extensively considered from an algorithmic approach, but mostly in terms of generative improvisation strategies (e.g. [18, 19]) and in relation to computational musicological analysis, including generative music theory [20] and corpus-based analysis [21, 22].
2. **Algorithmic generation of music notation**. Music notation generation is a complex issue, both from a theoretical perspective [23] and in terms of available, viable solutions related to specific projects. The Unreal Book project aims at investigating this issue in a broader sense, including all visual elements of the score;
3. **Formalization of jazz techniques**. Jazz theory has moved over the decades (in particular from the '60s) from a state of total absence (as it was implied into practices) to an abundant literature, related to the ongoing institutionalisation of jazz, discussing a variety of topics: mostly improvisation and harmony but also arrangement and practice routine. On a negative side, it has been observed that the quantitative increase in jazz pedagogy has led to a sort of homogenization. On a positive one, many resources are available, some detailing technical aspects, other discussing the latter in relation to historical developments, other proposing innovative approaches (see in general [1–3, 11]).

The Unreal Book project tries to match the Real Book's main features:

- dimension: the Unreal Book actually includes 102 pieces, notated in 1-page lead sheets (the most typical piece size in the Real Book). It is open to further expansion;
- heterogeneity: at the moment, seven different techniques have been used to compose lead sheets (see later), but the projects is meant as an open directory that allows to include other options and cross-hybridization among technical aspects from previously used techniques;
- visual consistency: not only the Unreal Book adopts the Real Book format, but it aims at mimicking its main visual features. Apart from cover design and TOC style, these include music notation typeface, page organization, title style. This aspect is not only related to visual aesthetics: rather, it helps musicians to enter a "Real Book mood" while approaching it. [1]

4. NOTATION ISSUES

In a symbolic algorithmic composition system, the score might be considered as the output of a terminal module that

[1] This is not at all irrelevant for music performance. For the very same reason, the legal Hal Leonard reissue of the Real Book [13] is marketed with the claim "You won't even notice the difference: [. . .] the covers and typeface look the same".

maps generated music data structures onto music notation symbols. From this perspective, music notation is meant as a form of data visualization. This encapsulation is not always possible. Historically, there is a feedback loop from notation to composition, as the former strongly constraints the latter [23]. In algorithmic composition practices there is a continuum going from integrated approaches, in which the whole pipeline from abstract music data processing to music notation is completely automated, to assisted or aided composition, in which the composer and the computer are loosely coupled – the computation agent providing elements (data structures, notation sketches) that are finally integrated into the score by the composer [24]. As in the Unreal Book project the notation format is explicitly given in advance, the most relevant issue is a technical one: for an output in the hundreds, it is mandatory to define at least partially automated strategies to include notation generation in the composition system. Many computer-aided composition packages (like OM [25], PWGL [26], Common Music [27], more recently Bach [28]) provide notation facilities that allow for drafting notation to be then finalized in a specialized environment. A viable option for totally automated notation generation in the style of the Real Book is the Lilypond notation software, a TEX-based language that compiles textual source files into PDF files [29]. Being text based, source files can be generated algorithmically. Lilypond source files can include not only melody notation but also chord symbols placed on top of it. Visual style can be adapted ingeniously so to reproduce all the main Real book visual items [2] . While elegant, this integrated solution does not allow an explorative approach to the results. As the output is a graphic file, there is no audio feedback associated to notation. MIDI commands can be included into Lilypond source but the resulting MIDI file must be open in a DAW environment with no interaction with the score. Moreover, MIDI does not support chord notation. Interactive features are instead typically provided by standard WYSIWYG notation environments such as Finale, Sibelius, Dorico, MuseScore. On the other side, these softwares use proprietary file formats. While not a notation format, MIDI has thus been used as a good compromise interchange solution (like in [22]). On one side, the target notation for standard lead sheets is metrically simple (resolution is limited to eight notes, rarely to 16ths, with only triplets as irregular groups) and can be accurately imported. On the other side, it is not possible to include chord symbols into MIDI, that must be consequently added by hand to the score, but they are comparatively sparse with respect to other notation symbols. The use of a WYSIWYG software allows for an immediate aural evaluation of results, and an eventual fine tuning of notation. MusicXML [30] is gaining momentum as an interchange format. A promising solution to be explored is to directly generate MusicXML code, integrating notes and chords, e.g. as allowed by the music21 toolkit [3] .

[2] See A. Lee, "Mimicking the Real Book", `http://leighverlag.blogspot.com/2015/12/mimicking-real-book-look.html`

[3] `https://web.mit.edu/music21/`

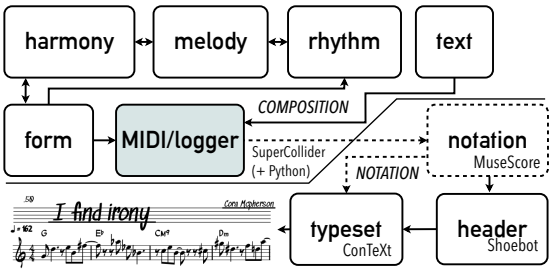


Figure 4. Overall system organization.

5. OVERVIEW OF THE SYSTEM

Figure 4 shows the overall software organization of the system. It can be divided into two subsystems: COMPOSITION and NOTATION. The COMPOSITION subsystem is developed in SuperCollider [31], and partially in Python (see later). It features four modules, meant as open libraries of functions for data processing and generation. Contents of the modules will be discussed in the next section in relation to examples. The rhythm module contains functions to generate metrically based events. The final format is always `[n, att, dur]`, where `n` is a placeholder to be filled with pitch information provided by the melody modules, and `att` and `dur` represent attack and duration, expressed in quarter beat units. The melody module includes various functions to generate pitch models: it outputs the same format of rhythm while replacing `n` proxies with actual values in MIDI notation. The harmony module is responsible for harmonic generation, that is, it outputs sequences of chords in standard alphanumerical notation. The form module coordinates the previous three modules. For each piece, it generates a melody and a harmony background. It also defines a general form in terms of sections (e.g. the classic American song form AABA) and takes care of handling section durations. The form module defines "composition configurations", that is, selections and parameterizations of functions from the various modules, including a generated bpm tempo. Logical ordering of the modules depends on the composition configuration (hence the double arrows between modules in Figure 4). The MIDI/logger module writes melodies into MIDI files, including tempo and key signature. When key signature is not decided in advance, it is inferred by comparing altered pitches in melody with various key signatures, and taking the signature that requires less momentary alterations. Chords cannot be included into a MIDI file, and are written by MIDI/logger into an easily human readable text file. The form module is typically used iteratively, feeding the MIDI/logger with 100 pieces in one shot, in relation to a certain composition configuration. Composition parameters are setup in in the SuperCollider/Python code. A crucial aspect of the Real Book is to be a collection of works written by many composers. The text module is devoted to generate names of composers and title of pieces. For each composition configuration, a batch of 5 composer names is generated: these groups of "composer" labels thus represent a specific

"style". Each composer is then associated with 20 titles. Composer names are generated from lists of most popular names (male and female) and surnames in the USA [4]. This is both a homage to the birth of jazz in New Orleans, and a good strategy to introduce variety, as USA have a largely differentiated linguistic community. Each title is generated by selecting one of the 50k movie reviews from the Large Movie Review Dataset [5], and then extracting and processing a textual fragment. The database has been chosen as it provides a colloquial tone typically associated with jazz titles, while paying a homage to cinema, an art form which has developed in 20th century, like jazz.

The NOTATION subsystem has the notation module at its core. The notation module is operated manually (hence the dashed contour in Figure 4), by interacting with the MuseScore notation editor. The latter is used to import MIDI files, and to edit them. MuseScore's importing options may be used to initially tune the notation. If occurring, editing involves notation aspects (e.g. F♯ might be replaced by G♭, da capo symbols might be inserted) but not music content. Chords are added by hand by taking as a reference the harmony log file relative to the MIDI file. The notation module is where the evaluation step takes place. Each piece, once imported is evaluated by the composer on the base of a set of loosely defined criteria, such as rhythmic complexity (pieces with too simple/complex rhythms are discarded), melodic contour (too static/variable melodies are discarded), melody/harmony clashing, etc. This manual filtering operation has resulted in an acceptance rate ranging between 10% and 20% of the generated pieces for the different composition configurations. While centered around acceptance and final notation of the pieces, the evaluation step provides also feedback on composition configurations, so that it can lead to various modification in the strategies on which these have been based. After acceptance and editing, pieces are exported in PDF files named with the format `composer_title`. The remaining part of the process is again fully automated. By using the Shoebot vector graphics package [6], a Python script is used by the header module to create automatically for each piece a graphic file containing the title, the composer and a progressive page number based on alphabetical ordering of the pieces, in the style of the Real Book. The header module also generates a source file for ConTEXt, a TEX-based typesetting document system [7]. The source file, including references to all files, is compiled into the final Unreal Book PDF. In terms of Computer-aided composition systems, the architecture can be thought as "fluid", i.e. made up of various modules "glued" together by two high level programming languages (SuperCollider and Python) [24].

6. SEVEN EXPERIMENTS

In the following, seven composition configurations of the previous system are described. These configurations, each of which associated with a set of composer labels, are meant as formalizations based on various features typically associated with jazz composition. Techniques are inspired by harmony/melody relationships as discussed in literature, mostly on improvisation [8]. In fact, there is a substantial permeability between composed melodies and improvised ones, as many times the latter are turned by jazz players into composed motives [11].

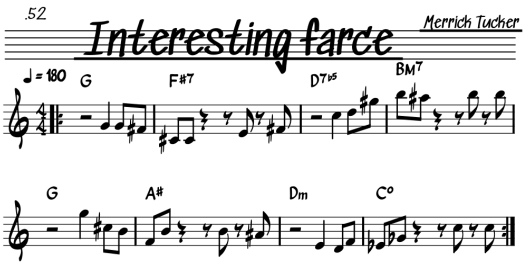


Figure 5. Atonal boppers example.

6.1 Atonal boppers

As the name suggests, Atonal boppers refers to both atonality and bebop. From the rhythmic point of view, bebop style results in less legato, with a distinctive presence of fast eight notes and less syncopations across bars [1]. In particular, Thelonious Monk's music is characterized by shifting riffs and accents, and isolated notes [6]. Atonal boppers are thus loosely inspired by these features. In this case, rhythm is handled by a drum machine-like pattern generator (contained in the rhythm module) that can be tuned so to generate blocks of a given metric duration with a certain density (i.e. average number of events for time unit). There are no slurs across bars. Figure 5 shows the section A of a piece. Here the rhythm pattern has a 2-bar duration, and is then repeated for the whole section. This repetition is crucial to ensure a certain degree of redundancy as the melody is freely atonal, a feature occurring in more experimental bebop pieces. In order to fill the rhythm pattern, the melody module exploits a Brownian generator: starting from a pitch and given a certain range in terms of semitones (e.g. ± 3), it generates a new pitch, then the process is reapplied. The Brownian model is interesting as it creates pitch contours. If a new pitch is outside a given overall boundary (substantially the treble staff, as it is customary), then it is flipped by 1 octave so to stay inside. The harmony module then provides automatic harmonization of the melody, in three steps. First, as the chords change at every bar, for each bar all the pitch classes in the melody are collected in a set, and stacked by thirds. Then, the resulting set is matched against a collection of given chords in a normalized form (e.g. major triad = [0,4,7]). Each chord is ranked in relation to how many pitches it is able to match in the pitch set, and the best is taken. Finally, in case more chords are available with the

[4] https://namecensus.com/

[5] https://ai.stanford.edu/~amaas/data/sentiment/

[6] http://shoebot.github.io/shoebot/

[7] https://wiki.contextgarden.net/Main_Page

[8] Code for the composition subsystems is available here https://github.com/vanderaalle/unrealBookComp

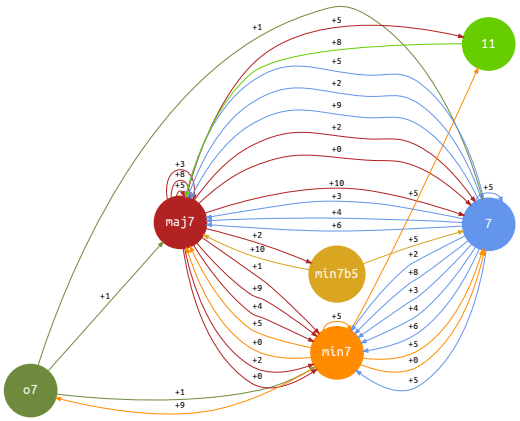


Figure 6. Harmonic graph.

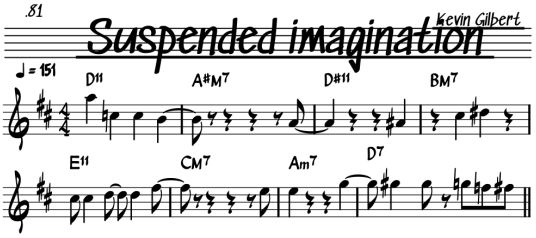


Figure 7. Forgetful harmonizers example.

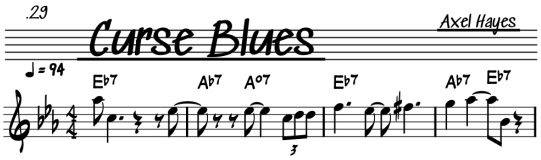


Figure 8. Bluesers example.

same score, one is selected on the base of a given priority ranking, that is: chord types have been ordered so that e.g. major chords have a higher rank than diminished ones. In this case, harmonies are limited to 7th chords (as can be seen in Figure 5). The form module randomly chooses among various section structures: AA, AB, ABA, ABAB, ABAC, and each section can have a length of 4, 6, 8, 12 bars. Figure 5 shows the A section of the piece, with a standard 8 bar length.

6.2 Forgetful harmonizers

Forgetful harmonizers are obtained by reversing some assumptions of Atonal boppers. Here, the rhythm module exploits a "time tape" model: time units in quarter notes are subdivided in 16th notes, and the latter are randomly grouped into longer events (triplets included). Then, some of the events are deleted according to a density parameter, thus creating rests. This process yields to irregular rhythmic groups. The relationship between melody and harmony is reversed, as the latter comes first in the generation process. Chosen chord types (mostly 7th chords) are organized in a graph in which each chord type (e.g. `min7b5`) is associated with a list of successors, based on [5] (Figure 6). As chords are taken into account, pitch classes, represented as integer in the range $[0...11]$, are at stake. Each edge is labeled with the number of positive semitones to reach the new pitch class root (e.g. +5 is the relative subdominant) from the chosen starting one. As chords are taken into account, the resulting pitch is to be taken $\mod 12$ (pitch class). In short, a cyclic graph results, that can be traversed randomly (and starting from a random vertex), each path representing a chord sequence once a starting pitch class is given. Actual pitches are inserted into the rhythmic pattern by looking at the relative harmony (again, one for each bar). A set of modes is available, and each chord type (e.g. `min7`) is associated with a subset of relative modes (e.g. minor, phrygian, dorian). A Brown process picks up a random pitch and selects a mode relative to the actual chord: if the pitch is in the mode, it is taken as is, else it is matched against the nearest one in the mode (e.g. in the context of Cmaj an E♭ becomes an E given a lydian mode). The next pitch is chosen randomly in a settable range around the previous pitch (e.g. ± 3 semitones), and again matched against one of the available modes for the actual chord. If a pitch duration extends across the bar, then the pitch is matched against the intersection of two modes, one for each chord. Form is organized as in the Atonal boppers case. This configuration is based on two assumptions. First, in jazz many times harmony comes first, not only in improvisation but also in composition, as in the cases when a given chord sequence is reused for a new piece (see the Rhythm Changers subsection). Second, since bebop there is a strong relation between harmonies and modes/scales (see [4] for an extreme application). In Forgetful harmonizers, harmonic sequences and harmony/mode relationships are based on [5]. Figure 7 shows an example, section A.

6.3 Bluesers

Bluesers are inspired by the classic 12-bar blues form. In this case, form is fixed a priori, and harmony is generated by reading an annotation file including typical 12-bar blues chord progessions [9]. Thus, in this case form and harmony are coupled and come first. Rhythm is again generated via the time tape model. As in the case of Forgetful harmonizers, melody matches harmonies, this time exploiting typical blues modes (e.g. including blues minor and bebop dominant scales [5]). Figure 8 shows an example with (matched) harmonies changing every half bar. As a side note, the typical "Blues" suffix has been appended to most titles.

6.4 Parkerians

Parkerians follow a very different path. Historically, the Charlie Parker Omnibook [32], a collection of 50 tran-

[9] E.g. `https://en.wikipedia.org/wiki/Twelve-bar_blues`

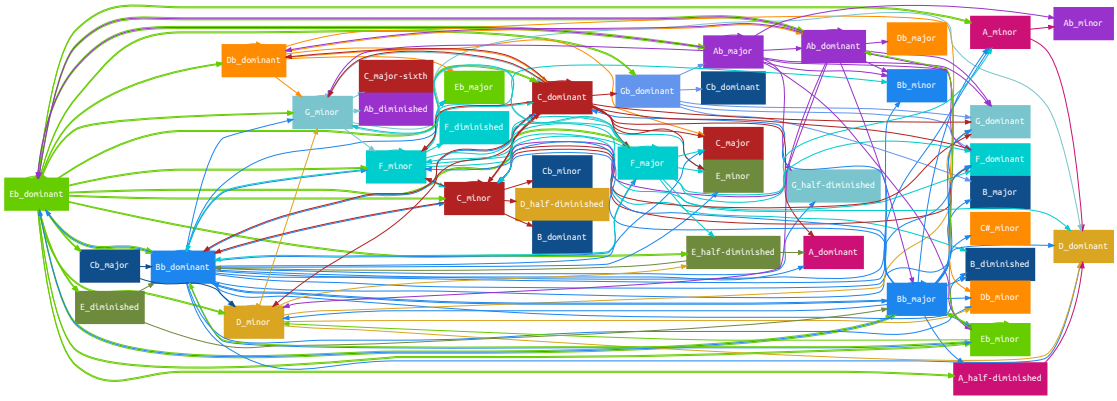


Figure 9. E♭ dominant harmonic subgraph from Charlie Parker's Omnibook.

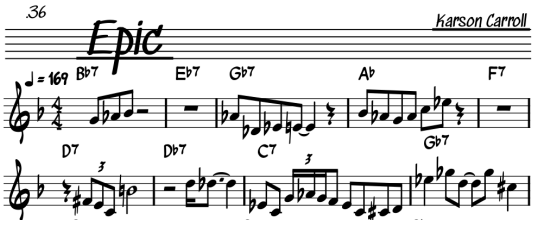


Figure 10. Parkerians example.

scriptions of Parker's solos, has been a crucial source of inspiration for generations of jazz players. The Omnibook has been annotated digitally [18] and is freely available in various formats (MIDI, MusicXML, MuseScore) [10]: the repository includes a Python script that allows to extract all the MusicXML data from each tune, that is, time-stamped melodies and chords. In Parkerians, following the model of Forgetful harmonizers, first a data structure has been created, in which each chord (in this case retaining the pitch class, e.g. B♭) is associated to all its chord successors in the whole corpus. The graph in Figure 9 is a subgraph of the overall, cyclic harmonic graph, showing for sake of readability only the 3-chord sequences starting with E♭ dominant. Enharmonic pitch classes are shown with the same color, regardless of their chord quality, and arrow colors are related to the starting pitch class. A second data structure is then generated that associates for each chord all the melodic fragments that are relative to it in the corpus. These two data structures bind together rhythm, melody and harmony. The form module selects a form among various options (AA, AB, ABA, ABAB, ABAC), and for each section a duration in bars (like in Atonal boppers). Then, for each section a chord sequence is obtained by traversing the harmonic graph. Finally, for each chord in the generated sequence, a melodic fragment is randomly choosen from those relative to the chord in the corpus. The whole set of compositions is thus a Markovian, harmony-driven recombination of Parker's melodic fragments, poured into a new form. An example is shown in Figure 10.

6.5 Modalists

Modalism in jazz typically refers to a variety of solutions based primarily on one or more modes at the basis of a piece. This usually results in a slower harmonic rhythm and in the use of one or more modes specified in advance as a pitch reservoir. A chord acts as a wider background for a mode-base melodic block. In Modalists, a set of modes is taken into account. Initially, a chord is generated from a degree in the chosen mode by stacking thirds. Then, the following chord is generated from a mode having at least one pitch in common with the starting one. Chords last 4 bars. As in Parkerians, rhythm and melody are extracted from existing sources. The Jazzomat project [11] makes freely available a large corpus of solo transcriptions, covering a variety of performers, styles, epochs [22]. The database can be explored by means of the MelospyGUI interactive tool, and selected items can be exported in MIDI. For Modalists, compositions from Jazzomat database have been chosen, converted into MIDI, re-edited so to simplify complex rubato rhythmic grouping, finally re-exported in MIDI. As solos are typically very dense, durations have been doubled. Then, all resulting melodies have been sliced into 1-measure-duration fragments. The recombining process is based on an enlarging reservoir. The reservoir is filled at initialization with a random fragment, then, every time a fragment is picked up, a new fragment is added to the reservoir. In this way, the reservoir keeps memory of the available fragments, leading to repeating patterns in the generated melody. Once selected, fragments are adapted to the mode occurring in their context, like in Forgetful harmonizers. In the example given in Figure 11, the same initial 2-bar melodic/rhythmic fragment is repeated thrice, then a new one is presented. Once the harmony changes at bar 5 from Fo7 to EM7, the recurring fragment is adapted to the different mode. The form module simply defines a variable number of 4-bar chord units.

[10] https://nubo.ircam.fr/index.php/s/BPtsmcqPQCNedgc

[11] https://jazzomat.hfm-weimar.de

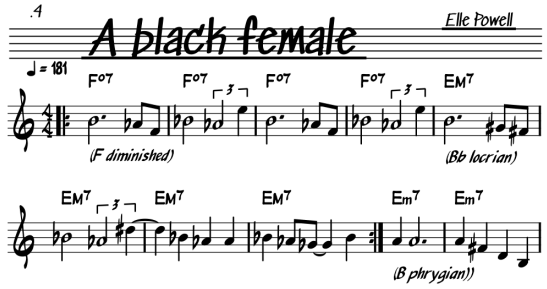


Figure 11. Modalists example.

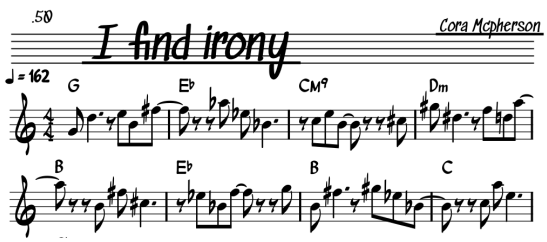


Figure 12. Minmaximalists example.

In terms of notation, modes are logged with harmony so to be reported –as customary with modal pieces– into the final score, as can be seen in Figure 11.

6.6 Minmaximalists

While rooted in blues and in post-impressionistic harmony, jazz has soon incorporated a variety of techniques developed in different contexts. Contemporary music techniques, above all serial and twelve-tone procedures, have been widely explored, in particular by the so-called Third Stream movement [2,3]. Inspired by this perspective, Minmaximalists are based on a serial, but not twelve-tone, technique. Variable length series between 3 and 6 elements are used for the pieces. Each integer item from the generated series is mapped onto a duration while some eight-note rest is added, thus defining the rhythmic pattern to be repeated. Analogously, a mapping strategy converts the same values from the series into pitches. At each repetition of the rhythmic pattern, the pitch series is then transposed following a random interval pattern. This organization results in a combination of rhythmic redundancy and melodic variety. In Figure 12 the rhythmic/melodic pattern has a duration of 6/4, so that two patterns fill 3 bars. Harmony advances at a regular 4/4 pace, thus providing a second rhythmic layer. Chords are obtained by adopting the same strategy used for Atonal boppers, but in this case harmonies are more complex, up to 9ths rather then 7ths. Coherently with the experimental assumptions, the form, while maintaining a standard overall duration of 32 bars, is not the classic AABA. Section A can be 8-bar, to be repeated (then AA) or a single 16 bar. Section A' is the same of A if the latter has a duration of 8 bars, or half A if the duration of A is 16 bars. The resulting pieces are atonal but the melody is anchored to jazz harmony, while rhythmically they are based on a straight 8-note rhythm, with a certain geometrical flavor, thanks to the eight-note based patterns shifting on the 4/4 bar grid. Hence the reference both to Minimalism and to its serial opposite.

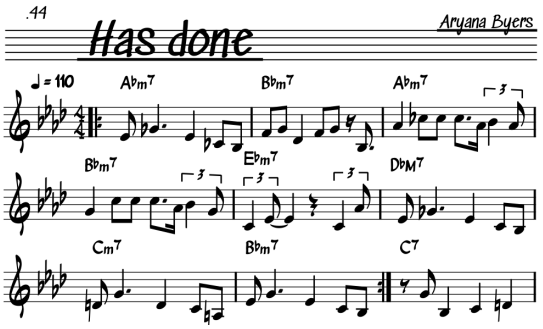


Figure 13. Rhythm changers example.

6.7 Rhythm changers

Some chord progressions are widely used in jazz. The most famous is derived from Gershwin' song *I Got Rhythm*. Known as Rhythm changes [5], it is at the base of many famous jazz pieces. Rhythm changes are thus taken into account as the starting point of the composition configuration. Thus, the form is the standard 32-bar AABA' from *I Got Rhythm*. In Rhythm changes, the harmonic rhythm proceeds at 2 chords per bar in the section A, while slowing down at 1 chord every 2 bars in section B. A common practice in jazz is to perform chord substitutions, that is, to replace original chords with new ones. This is seen as instrumental to offer new melodic possibilities. As discussed by Liebman [33], chord substitution can also be seen as a way to redefine harmony on the fly while improvising: a certain given chord is thus mentally replaced by the improviser with a different one, the latter acting as a reference for expanded melodic construction. In Rhythm changers, first, a second set of chords is defined for section A, which is an altered/complexified version of the original sequence: as an example, the starting maj7 chord is replaced by a min9. Section B is subject to an extensive chord substitution, following the so-called Coltrane changes, while, as a second step, it is altered/complexified as in A. Also, the harmonic rhythm is converted into 2 chords per bars, like in A. These richer A and B harmonies are used to create modes. For each bar, the union of the two chords' pitch sets defines a reference mode. The raw material from melody is created from Jazzomat, as described for Modalists. The fragments are matched onto the reference mode for their relative bar. While chord substitutions are used to create modes, they are not displayed in the score. Rather, the original harmony is displayed for section A, but only in the first chord of each bar. Analogously, section B displays the Coltrane changes but before the complexification step, and only the first chord of each bar. In short, harmonic complexification results in a surface harmonic slower and homogeneous rhythm, while feeding the melodic construction. An example (in A♭) is shown in Figure 13.

7. CONCLUSIONS AND FUTURE WORK

The Unreal Book project has proven to be an interesting test bench to develop an algorithmic-based, computer-aided composition system capable to integrate final typesetting by means of a "fluid" architecture. Performances of pieces are planned, so that results can be tested and properly evaluated in the context of jazz playing. The project can be expanded by implementing new composition configurations that may be triggered both by further investigations in jazz theory and analysis, and by various algorithmic composition processes. In particular, larger harmonic contexts could be taken into account to ensure harmonic structure, and more automated data extraction procedures could be implemented.

8. REFERENCES

[1] T. Gioia, *The History of Jazz*. Oxford: Oxford UP, 2011.

[2] S. Deveaux and G. Giddins, *Jazz*. New York: Norton, 2009.

[3] S. Zenni, *Storia del Jazz. Una prospettiva globale*. Viterbo: Stampa alternativa, 2012.

[4] M. Levine, *The Jazz Theory Book*. Petaluma: Sher Music Co., 1995.

[5] R. Rawlins and N. E. Bahha, *Jazzology*. Milwaukee: Hal Leonard, 2005.

[6] S. Zenni, *I segreti del Jazz*. Viterbo: Stampa alternativa, 2007.

[7] B. Bauer, Ed., *The New Real Book*. Petaluma: Sher Music Co., 1988.

[8] D. Haerle, *The Jazz Language*. Miami: Studio 224, 1980.

[9] A. Jaffe, *Jazz Harmony*. Tübingen: Advance Music, 1996.

[10] J. Mulholland and T. Hojnacki, *The Berklee of Jazz Harmony*. Boston: Berklee Press, 2013.

[11] P. F. Berliner, *Thinking in Jazz. The Infinite Art of Improvisation*. Chicago: Chicago UP, 1994.

[12] T. Gioia, *The Jazz Standards. A Guide to the Repertoire*. Oxford: Oxford UP, 2012.

[13] *The Real Book*. Hal Leonard, 2004.

[14] G. Nierhaus, Ed., *Patterns of Intuition*. Dordrecht: Springer, 2015.

[15] A. McLean and R. T. Dean, *The Oxford Handbook of Algorithmic Music*. New York: Oxford UP, 2018.

[16] C. Roads, *The Computer Music Tutorial*. Cambridge, MA, USA: MIT Press, 1996.

[17] G. Nierhaus, *Algorithmic Composition. Paradigms of Automated Music Generation*. Wien: Springer, 2009.

[18] K. Déguernel, E. Vincent, and G. Assayag, "Using Multidimensional Sequences For Improvisation In The OMax Paradigm," in *13th Sound and Music Computing Conference*, Hamburg, Germany, 2016.

[19] D. Vassilakis, A. Georgaki, and C. Anagnostopoulou, "'Jazz Mapping' an Analytical and Computational Approach to Jazz Improvisation," in *Proceedings of the 16th Sound and Music Computing Conference (SMC2019)*, 2019.

[20] M. J. Steedman, "A generative grammar for jazz chord sequences," *Music Perception*, vol. 2, no. 1, pp. 52–77, 1984.

[21] Y. Broze and D. Shanahan, "Diachronic changes in jazz harmony: A cognitive perspective," *Music Perception*, vol. 31, no. 1, pp. 32–45, 2013.

[22] M. Pfleiderer, K. Frieler, J. Abeßer, W.-G. Zaddach, and B. Burkhart, Eds., *Inside the Jazzomat*. Mainz: Schott Campus, 2017.

[23] A. Valle, *Contemporary Music Notation. Semiotic and aesthetic aspects*. Berlin: Logos, 2018.

[24] ——, "Integrated Algorithmic Composition. Fluid Systems for including notation in music composition cycle," in *NIME 2008: Proceedings*, 2008, pp. 253–256.

[25] G. Assayag, C. Rueda, M. Laurson, C. Agon, and O. Delerue, "Computer-assisted composition at IRCAM: From PatchWork to OpenMusic," *Computer Music Journal*, vol. 23, no. 3, pp. 59–72, 1999.

[26] M. Kuuskankare and M. Laurson, "Expressive Notation Package," *Computer Music Journal*, vol. 30, no. 4, pp. 67–79, 2006.

[27] H. Taube, "An introduction to Common Music," *Computer Music Journal*, vol. 21, no. 1, pp. 29–34, 1997.

[28] A. Agostini and D. Ghisi, "A Max Library for Musical Notation and Computer-Aided Composition," *Computer Music Journal*, vol. 39, no. 2, pp. 11–27, 2015.

[29] H.-W. Nienhuys and J. Nieuwenhuizen, "LilyPond, a system for music engraving," in *Proceeding of the XIV CIM 2003*, Firenze, 2003, pp. 167–172.

[30] M. Good, "Lessons from the adoption of MusicXML as an interchange standard," in *XML 2006 Conference Proceedins*, D. Megginson, Ed., 2006.

[31] S. Wilson, D. Cottle, and N. Collins, Eds., *The SuperCollider Book*. Cambridge, Mass.: The MIT Press, 2011.

[32] J. Aebersold, Ed., *Charlie Parker Omnibook*. Atlantic, 1946.

[33] D. Liebman, *A Chromatic Approach to Jazz Harmony and Melody*. Advance Music, 2015.

SKETCHINGSONICTRAJECTORIES
A IANNIX TOOL FOR COMPOSING THE ELECTROACOUSTIC SPACE

Julian SCORDATO (julian.scordato@conservatoriopollini.it)[1], **Nicola PRIVATO** (nicola.privato@gmail.com)[2], and **Nicola RACCANELLI** (raccanelli.nicola@gmail.com)[2]

[1]*SaMPL - Sound and Music Processing Lab / Department of New Technologies and Musical Languages*, **Conservatory of Padua**, Italy
[2]*Department of New Technologies and Musical Languages*, **Conservatory of Padua**, Italy

ABSTRACT

This paper provides a report on the development of SketchingSonicTrajectories (SST), a software tool for sound spatialization that simplifies the user interaction in a specific usage of IanniX graphic sequencer: SST integrates multi-track audio playback, Ambisonics spatial audio processing, and IanniX score and performance management.

In addition to a project description and details on the user interface design, several strategies for the composition of the electroacoustic space with SST will be proposed, also with reference to the score examples included in the software package. The suggested exemplification is intended to illustrate the current software features and guide the user in the design and articulation of advanced movements of sound sources to achieve a multi-channel sound projection without getting entangled in software technicalities.

1. INTRODUCTION

Over the years, several panning techniques such as WFS [1], Ambisonics [2], and VBAP [3] have been developed with the aim of simulating the position and movement of sound sources in space accurately, and different approaches on their use in design and compositional practices have been adopted [4]. From the user side, strategies for sound spatialization commonly foresee automations on multitrack sequencers including software plugins, algorithms developed on programming environments, and performative practices involving a mixing console or a controller. In addition to standalone applications, certain implementations of spatialization algorithms can be integrated in software environments for real-time sound synthesis and processing (e.g., Max/MSP, Pure Data, or SuperCollider) and digital audio workstations. Among the notable tools available for free are Ircam Spat [5], ICST Ambisonics [6], HOA Library [7], Zirkonium [8], and SpatGRIS [9]; these offer a high degree of customization. However, it is worth mentioning that Zirkonium and SpatGRIS are currently incompatible with Windows, while spatialization libraries such as Spat, ICST and HOA depend on programming environments for their implementation, thus requiring specific skills from the user. Software plugins that rely on multitrack sequencers have the advantage of an easier approach, but limited control over the design of spatial movements; their representation is often trivial consisting in the variation of single parameters versus time. In different ways, these tools face the difficulty of interfacing the user with the notation and the representation of space as a compositional parameter.

In the ecosystem of sound spatialization tools, SST arises from specific questions: how to find a handy way to formalize and reproduce spatial parameters from the micro to the macro-form, in relation to the audio content; how to facilitate the design of sound trajectories for multi-channel electroacoustic works and sound design projects; how to offer flexibility to different live / studio contexts and broad compatibility with software and operating systems. The goal was to give access to a potentially wider user community by developing a tool capable of offering multiple approaches to notation and performance, and to bring attention to design and artistic issues instead of retaining the user on technical and programming aspects.

We chose to consider the Ambisonics equivalent panner included in the ICST Tools[1] for its flexibility in terms of parameter customization, suitability to a wide range of contexts, and licensing under the Revised BSD License. In particular, the uncoupling of the spatialization project from the electroacoustic configuration makes the system adaptable to different speaker setups, with the only requirement of having the same kind of speakers preferably arranged in a regular angular way. In real-life scenarios, loudspeaker amount and placement are usually different in relation to the host organization's choices and other factors.

As for the graphic design features, we decided to use IanniX[2] and discard the objects included in the ICST package. IanniX proposes a poly-temporal and multi-formal open-source sequencer also useful for the creation and the performance of message-emitting control scores for sound spatialization [10]. Through the basic functions of the IanniX GUI, it is possible to notate and store various data such as the 3D position of sound sources and speakers, to design custom spatial movements and patterns with

[1] https://www.zhdk.ch/forschung/icst/software-downloads-5379

[2] https://www.iannix.org/en/whatisiannix/

precise event timing, and to visualize a clear representation of global behavior in time and space. In addition to the GUI, the OSC and JavaScript approaches to score can be combined together to define the path of various types of curves available in IanniX (freehand, linear, Bézier, parametric and circular curves) [10]; this can significantly facilitate the spatialization design process according to the different abilities and needs of users. Most of the tools available for spatialization design provide one-way and single-strategy approaches, flat and non-reactive representations of the current position and level of sound sources, and simple and stereotyped motions such as rotation, translation, and random walk. In some cases, trajectory descriptions must be provided in a breakpoint format (time, coordinates) that forces the user to a clunky approach, especially when designing a macro-form.

In order to interface the IanniX environment with the spatialization algorithm, we have developed a macOS and Windows application that also includes a multitrack reproduction device. The SST GUI was intended to facilitate the interaction with IanniX, guiding the user in a step-by-step project definition, proposing advanced customizable models of sound trajectories, allowing reactive and bidirectional management of the spatialization project, and avoiding dependence on DAWs. However, the user is given the option to choose the preferred audio input from external hardware or third-party software (via virtual audio drivers), which makes SST suitable for both studio and live projects.

After its first iteration [11], SST is currently under development at the Department of New Technologies and Musical Languages of the Conservatory of Padua. The software package is freely available for download[3]. An updated project overview accompanied by a description of the system components is presented in the Section "Project description". The user interface has been adapted for improved usability, adding IanniX score controls, transport synchronization, event handling and visualization. Through this, we aim to further facilitate the production and reproduction of a graphic score for sound spatialization (see "User interface design"). Several score examples are proposed and discussed in relation to their functionality, in order to demonstrate different procedures related to the use of SST and the included trajectory library (see "Compositional and performance strategies"). Additionally, the use cases within our Living Lab Music showcase have been taken into account (see "Use cases"). Finally, we conclude with some considerations and perspectives for the future development of the project (see "Conclusions").

2. PROJECT DESCRIPTION

The proposed system for the creation and reproduction of graphic scores for sound spatialization involves three main logical sections – SST, IanniX, and Mira – as described below. Communication is handled via OSC (to and from IanniX) and Bonjour protocol (between SST and Mira).

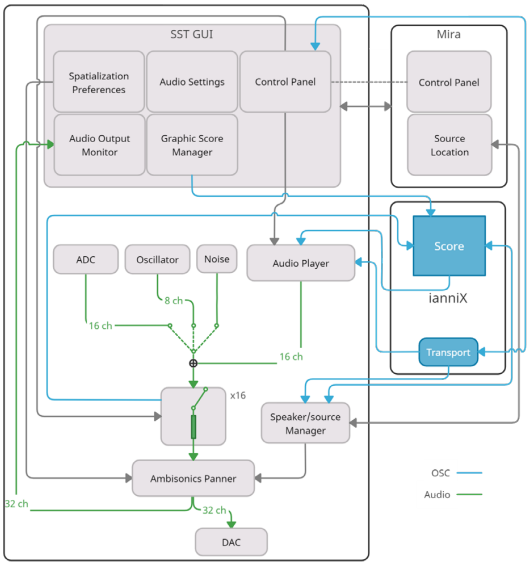


Figure 1. Block diagram.

2.1 SST

SST is an application developed in Max/MSP; it includes a Graphical User Interface (GUI) for managing the score and related audio content, a series of devices for interfacing with the *IanniX score* and *Transport*, and a signal processing stage for audio routing, playback and spatialization (see Fig. 1). The SST GUI provides access to several components: *Graphic score manager*, *Control panel*, *Audio settings*, *Spatialization preferences*, and *Audio output monitor*.

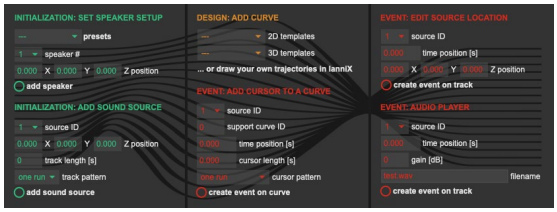


Figure 2. *Graphic score manager*.

The *Graphic score manager* defines the possible strategies for the creation and facilitated handling of objects within a IanniX score, aiming at the design of a spatialization project (see Fig. 2). It allows the user to set the speakers' number and position and to initialize the virtual displacement of audio signals and files into the ianniX score by organizing them into *sound sources* (see §3.2). Also, it suggests a series of 2D and 3D templates for the insertion of trajectory paths, and facilitates the interaction with the inserted objects by configuring their attributes and behavior as well as the visualization in IanniX of relevant information such as labels and audio level meters.

[3] https://www.julianscordato.com/projects.html#sst

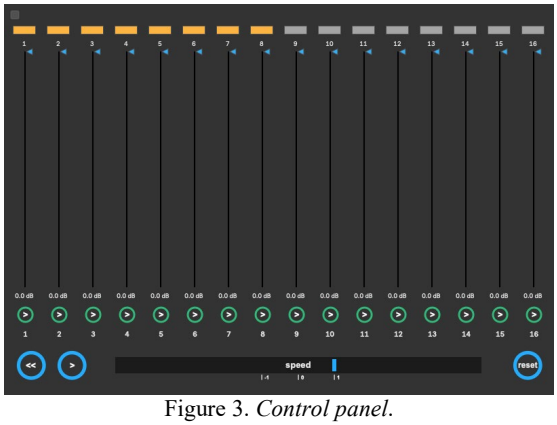


Figure 3. *Control panel*.

The *Control panel* can be accessed from both the SST application and the Mira app (see §2.3). It contains 16 potentiometers and playback buttons, one for each available sound source (see Fig. 3). This panel is synchronized with the *Transport* information in IanniX, giving the ability to combine score reproduction with real-time playback controls. Gain and mute controls for each audio channel can intervene at the input stage before the audio signal is routed to the *Ambisonics panner*.

The *Audio settings* module includes driver selection and audio I/O mappings. A further window is dedicated to the *Spatialization preferences* for adjusting the Ambisonics directivity and the amplitude attenuation pattern in the *Ambisonics panner*, which is based on the ambipanning~ object for Max/MSP [6]. The *Ambisonics panner* output is sent to the DAC as well as to the *Audio output monitor* in order to provide the user with visual feedback of the channel levels during score playback.

SST application offers the possibility to choose different types of audio inputs: *Audio player*, *ADC*, *Oscillator,* and *Noise*. The *Audio player* is responsible for the reproduction of the audio files assigned to each of the 16 available *sound sources* through the *Graphic score manager.* Activation of audio file playback with a desired gain level can be set in the score at any time position along a track. It is also possible to use the dedicated playback controls located in the *Control panel* (see Fig. 3). Except for the *Audio player*, the other inputs are mutually exclusive and can be switched via a selector; it is therefore possible, for example, to couple an external analog input with the playback of audio files using the same channel. The ADC can provide 16 channels from audio hardware as well as from virtual audio devices, in order to interface third-party audio applications with SST in real-time. To assist the user in testing the spatialization system, two ready-to-use audio inputs are selectable: a white noise generator and a polyphonic sawtooth oscillator. Available on channel 1, the noise generator can be conventionally used for a speaker test. The polyphonic oscillator, on the other hand, can be a useful tool in the design of spatialization models that involve multiple sources: it matches a single tone of the major scale to each of the sound sources, allowing the user to postpone the assignment of audio content.

Formatting of IanniX speaker and source position messages for the *Ambisonics panner* is done by the *Speaker/Source manager.* This device also turns on the audio inputs when the *Transport* starts and turns them off when score playback is stopped. In order to automatically update the configuration with any intervention on the score between two consecutive reproductions, the *Speaker/Source manager* forces IanniX to output the current status of all the elements present in the score following the *Transport* activation.

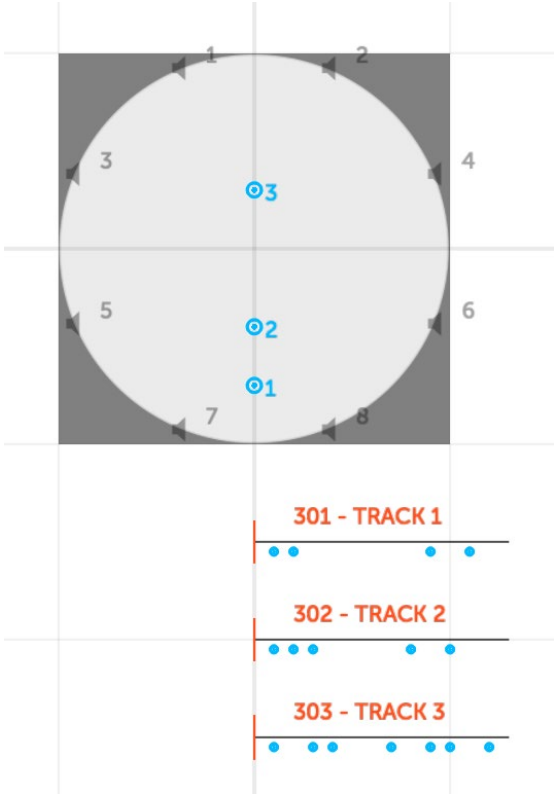


Figure 4. Example of static source positioning in a score included in SST software package (*sequence_1.iannix*).

2.2 IanniX

We have designated IanniX as the core sequencing engine and graphics platform for SST, due to its inherent poly-temporal and multi-formal sequencing capabilities and the flexibility it offers in terms of graphic notation [10]. For the description, two blocks are distinguished: the *Transport* and the *IanniX score* (see Fig. 1).

The *Transport* sends the score playback status to various devices: *Speaker/source manager*, *Audio player*, and *Control panel*. Available directly from the IanniX GUI as well as from the *Control panel*, the *Transport* represents the main control for the global reproduction of the graphic score and for the synchronization of routines such as input/output activation and source position refresh (see §3.3). Transport information is also sent to the *Audio player* in order to enable the reproduction of the audio files added to the project via the *Graphic score manager*.

The *IanniX score* stores the data and graphically represents the objects entered by the user through the *Graphic score manager*. The multi-formal representation space of the score includes a coincident reproduction of the Ambisonics 3D coordinate system [6] – which in turn can be considered as a convenient approximation of the electroacoustic space (see Fig. 6) – and up to 16 source-related automation tracks (see Fig. 4). The *IanniX score* acts as an interface that allows the user to intervene on the virtual position of speakers and sound sources, to manage the automation tracks and to set advanced attributes of the IanniX objects (see §4).

2.3 Mira

New to this SST iteration, this latter section consists of a touch-control interface for real-time source positioning which also provides global transport and audio playback controls. Developed in Max/MSP, this interface makes use of the Mira app for gestural input and visualization on a compatible mobile device [12].

Figure 5. *Source location* tab in Mira.

The proposed interface allows the user to combine the SST functionalities for score designing with the possibility of real-time intervention into the score during its performance; it contains two main tabs, both accessible via the Mira app: *Control panel* and *Source location*. Optionally accessible also from the SST GUI, the former provides audio controls for each channel (see §2.1). The latter allows real-time spatial manipulation of up to 4 selectable sound sources (see Fig. 5). Since the source position and playback controls can be set gesturally from a compatible mobile device, the user is free to move to the "sweet spot" while maintaining a certain degree of control over the score.

3. USER INTERFACE DESIGN

This section addresses the design of fundamental aspects of the SST interface for the purposes we have set ourselves (see §1). In the first subsection we will present issues related to the visualization of the elements in the score, from the macro-form to the micro-form, with a particular focus on the latter since in our approach to spatialization design we consider the macro-form as a result of the articulation of individual events that unfold over time. The second subsection will cover notational aspects, in particular how the score composition process was designed and how the IanniX objects were used for this purpose. Finally, we will address the aspects related to score reproduction and the possible performative approaches.

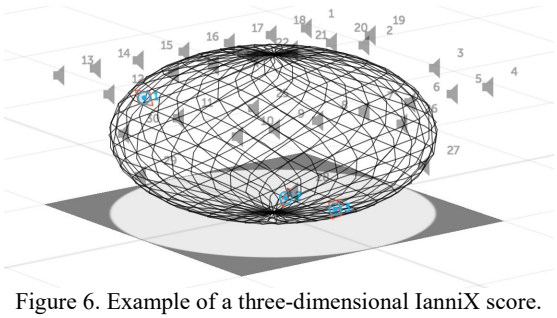


Figure 6. Example of a three-dimensional IanniX score.

3.1 Score representation

The three-dimensional Cartesian representation of the IanniX score in our specific application tends to be an approximation of a spatial configuration including both the actual speaker setup and the virtual positioning of the sound sources (see Fig. 6). The electroacoustic space is mapped according to the Ambisonics space definition [5], whose dimensions coincide with the absolute Cartesian coordinates of the rendering area of the IanniX score. A graphic texture has been placed to delimit conventional 2D boundaries. A similar two-dimensional representation is provided in the Mira interface for simultaneous control of up to 4 sound sources in live interventions (see Fig. 5).

Additionally, up to 16 source-related tracks are displayed in the lower area of the IanniX score (see Fig. 4). As in common sequencers, tracks provide additional timelines that normally serve the purpose of viewing and editing specific automations. In our application they constitute the support for the formal articulation of several types of events (see §3.2).

For the purpose of spatialization design, the content of a score includes a user-defined number of speakers, sources, and events notated as IanniX triggers; spatial trajectories are notated as IanniX curves; trajectory reading heads are notated as IanniX cursors with the function of driving the triggers associated with the same source.

During playback, the visualization in the main score area is synchronized with the events set along the tracks. Furthermore, to relate the audio content to the current position, a localized level meter is proposed for each active source by means of a HSV color variation in the related IanniX trigger.

3.2 Notation

In the design of the *Graphic score manager*, we distinguished three consequential phases in the user approach to creating a score: from the initialization of the electroacoustic setup (in terms of available sound sources and speakers), through the sketch of the sound trajectories, up to the addition of control events and audio content (see

Fig. 2). For the purpose of writing a spatialization score in SST, the basic IanniX objects (i.e. triggers, curves and cursors) take on certain functions.

In IanniX, a trigger is an "*object with the ability to send individual output messages*" [10]. Within the SST interface it can assume various connotations:

- *Speaker*, i.e. a graphical representation of the loudspeaker position with a custom texture (see Fig. 6); up to 32 speakers can be entered and displaced in a score;
- *Sound source*, namely an Ambisonics input (see Fig. 1) virtually positioned in the electroacoustic space; up to 16 sources can be displaced in the score; it is also possible to assign a single input to multiple sources through the I/O mappings included in the *Audio settings* window;
- *Event*: *Audio player* (see Fig. 2) creates a marker placed on a source-related track to start playing an audio file and set its volume level;
- *Event: Source location* creates a marker placed on a source-related track to set the three-dimensional displacement of a sound source in the score (see Fig. 4);
- any other trigger that causes an event in the score using a IanniX loopback message formatted as a command [13].

A IanniX curve is a "*graphical representation of a function or a vector-based path within the score*" [10]. Curves are used in the following cases:

- *trajectories*, i.e. the three-dimensional paths involving one or more sound sources located in the main score area (see Fig. 6);
- *tracks*, consisting of source-related timelines coupled with cursors aimed at activating events (see Fig. 4); in the SST GUI, the user can instantiate a track while initializing the related sound source (see Fig. 2);
- track *automations* that define the desired variation of up to two parameters as a function of time, considering the Y and Z axes (see Fig. 7).

A cursor in IanniX is a "*time-based graphical object that moves along the path of a linked curve and performs local and autonomous sequencing functions*" [10]. In the context of the SST user interface, cursors are applied for dynamic positioning of sound sources: using IanniX loopback messages, a trigger can be forced to follow the current position of a cursor; through this, it is possible to define the temporal behavior of a sound source in the score and make the source move along the path of a three-dimensional curve (see Fig. 6). Taking into account a modular temporal approach, multiple cursors can be associated to the same sound source to articulate different spatial-temporal behaviors in the perspective of a macroform. Secondly, IanniX cursors are used as part of source-related timelines for the activation of any colliding triggers (see Fig. 4), as well as for reading the values of any colliding curves (see Fig. 7).

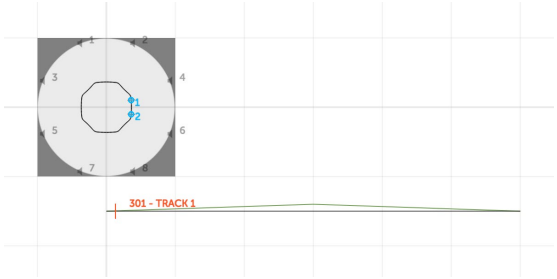


Figure 7. Example of track-level automation in a score included in SST software package (*supershape.iannix*).

3.3 Reproduction

As part of our goal of making SST suitable for both studio and live applications, we propose two main approaches related to the performance of a score. The former exploits the functions of the integrated *Audio player* (see §2.1), and is based on fixed audio content to be reproduced by means of events located on source-related tracks. The latter takes advantage of the ADC input for a more flexible configuration that allows the connection of external audio devices such as acoustic instruments, as well as other software and VSTs (using a virtual audio driver); this also led us to integrate the Mira interface into the system (see §2.3), in order to allow the user greater control over the audio content.

Once the score has been set properly, the audio inputs and outputs can be configured in the SST GUI from the *Audio settings* window. The audio input level can be checked on the meter incorporated in the *sound source* representation only after starting the score playback. The *Transport* (see §2.2) can be activated via the *play* button which is synchronized between IanniX and the SST GUI. The activation involves updating the status of the IanniX objects representing sources and loudspeakers, which are forced to send control messages to SST. Any active events or trajectory-related cursors will reposition the sound sources on the planned paths or position. When score playback is stopped, audio input routing is interrupted accordingly.

4. COMPOSITIONAL AND PERFORMANCE STRATEGIES

This section is focused on the score prototypes included in the software package with the aim of exemplifying some possible SST applications both on the basis of the *Graphic score manager* functionality and through a more advanced use of IanniX (e.g. including the loopback interface and JavaScript functions).

The trajectories library accessible in the *Graphic score manager* offers a potential starting point for a design process aimed at the dynamic spatialization of sound sources (see Fig. 2). Indeed, the proposed 2D and 3D curve templates rely on basic parametric equations that can easily transform and reshape the objects in IanniX and therefore adapt to the user's needs. The temporal behavior of a single trajectory, as well as the articulation of different movements in a macro-form is made possible by the

definition of cursor's parameters: *time position* [s] and *cursor length* [s]. The former establishes the starting point in relation to global transport in IanniX; the latter defines the cursor's duration [s] along its support curve. A more advanced approach may also foresee the use of cursor acceleration patterns, recursion, or JavaScript functions, as described below. The positioning of a *sound source* can also be achieved by means of a single-step movement, for example in the event of a preset change or initial position setting. This can be practically implemented through track-level automations or direct displacement of the related trigger in the IanniX score (see Fig. 4) as well as through a gestural approach using the Mira app (see §2.3).

Overall, we have identified some significant usage strategies, not with the intention of being exhaustive but rather to illustrate some modular paths that from the instance of individual objects proceed towards a potential complete project of sound spatialization. These are listed as follows:

- constant speed trajectory system with a predetermined and predictable space-time behavior resulting in a "control score" [10]; the cursors instantiated via SST can move either along a single curve or different curves intended as their spatial path; also, they can be temporarily synchronized or not, according to the cursor parameters set in the SST graphic score manager (see *solar_system.iannix* and *DNA.iannix*);
- trajectories with non-linear speed defined by cursor acceleration patterns; each cursor can behave in a specific and independent way within the score (see *circles.iannix*); advanced object attributes such as cursor acceleration can be set via the Inspector panel, which is part of the IanniX GUI;
- variable trajectories according to a parameter value defined by track-level automation (see *supershape.iannix*); as with traditional sequencers, the automations offer the possibility of defining the variation of mapped parameter values as a function of time (see Fig. 7);
- variable trajectories according to a colliding object that modifies the parametric equation defining the cursor path; automations and events can be set directly in the representation space of the score, thus exploiting the IanniX-specific three-dimensional and poly-temporal sequencing features (see *sphere.iannix*);
- variable trajectories according to themselves in a "recursive score" setting [10]; the IanniX loopback interface allows the user to route an output message to the IanniX input for a score command instance; a command can affect a single object as well as a group of objects, as in *spirals.iannix*;
- static source positioning by means of track-level automation (see Fig. 4); the source positioning event set via the *Graphic score manager* is designed in the score at the track level; this allows the user to have a view of the entire macro-form relating to single-step movements of the sound sources; the same approach applies to the triggering of audio files via the internal player (see *sequence_1.iannix*);
- static source positioning using random functions that produce a "generative score" [10]; track-level automations can be read in unconventional ways by going beyond the notion of linear timeline; different strategies are foreseen, such as the JavaScript implementation of a random function in a trigger-related output message that sets the object's position (see *random_points.iannix*) or the use of a time-related IanniX command [13] that affects the reading position on the timeline (see *sequence_2.iannix*);
- dynamic source positioning via gestural input using the Mira interface (see §2.3); this approach produces an "interactive score" [10] in which predetermined temporal behaviors reproduced in IanniX can be combined with live controls;
- creation of trajectories and performance control through other external inputs; specific OSC/MIDI input messages sent from third-party devices can be related to adding and controlling objects in a "reactive score" [10]; this requires a JavaScript approach to the *onIncomingMessage* method included in a IanniX score (see *touchosc.iannix*).

The operational proposals listed above can be combined functionally and conveniently in a score in order to produce more complex behaviors intended as articulations of simple events in the perspective of a macro-form (i.e. the spatialization project as a whole).

5. USE CASES

SST has been used extensively in an artistic and technological scenario of *Living Lab Music*[4], a SaMPL showcase that combines contributions from established artists and researchers with artistic products from the Department of New Technologies and Musical Languages of the Conservatory of Padua. Due to the restrictions of the COVID-19 pandemic, no audience could take part in the 2021 edition. However, binaural audio and video recordings were made with the aim of disseminating the content online, while still returning an acoustic image of the specific setup and venue. Living Lab Music 8 took place at the *Pase Platform*[5] in Venice, which provided a 30-channel speaker system whose near-hemispherical arrangement (see Fig. 6) was particularly suitable for advanced spatialization. We could observe that the ICST Ambisonics equivalent panning algorithm presented a satisfactory response after adjusting the parameters (Ambisonics order and level attenuation curves, in particular). A compromise was needed between the angular precision given by the Ambisonics order increase

[4] https://vimeo.com/sampllab/albums

[5] https://pase-platform.com

and the uniformity of the audio level in all points of space, which instead required a wider polar pattern.

Ranging from fixed-media works to multimedia performances, the applications of SST in this context covered both the approaches identified in §3.3 (sometimes used in combination) and all the strategies proposed in §4, allowing us to test the full operation of SST. Early stage electronic music students could easily implement SST in their stand-alone or interactive projects after a minimal training period, only occasionally encountering issues in the interaction with the IanniX score (see §6). However, thanks to IanniX's graphic sequencing capabilities combined with the ability to control the spatialization of sound sources in real-time via Mira or other interfaces, SST has proven flexible to the diverse performative and compositional needs of the students and artists involved.

6. CONCLUSIONS

Through SST we intend to propose an easy-to-use IanniX tool aimed at the design of multi-channel sound projections for musical works and sound design projects, but also to favor an approach to electroacoustic composition stimulated by the formalization of virtual sound trajectories organized in space and time. In a future version of the software we plan to focus on several additions and bug fixes, including:

- a reverb implementation for improved distance simulation and room treatment;
- advanced management of Ambisonics directivity using independent parameters for each sound source;
- better management in the instance of IanniX objects, which can currently generate bugs related to the programming logic of IanniX score files;
- an improved trajectory library and related score examples; we realize that we have not been able to consistently systematize a curve library, and have struggled to find satisfactory models in the literature to draw from;
- the possible inclusion in the score of a further representation of cursor-related events at the track level (see §3.2); in the articulation of different trajectories, it is currently not possible to have a view on the entire macro-form, as the cursors appear only during their movement.

7. REFERENCES

[1] A. J. Berkhout, D. de Vries, and P. Vogel, "Acoustic control by wave field synthesis," The Journal of the Acoustical Society of America, 93(5), 1993, pp. 2764-2778.

[2] D. G. Malham and A. Myatt, "3-D sound spatialization using ambisonic techniques," Computer Music Journal, 19(4), 1995, pp. 58-70.

[3] V. Pulkki, "Virtual sound source positioning using vector base amplitude panning," Journal of the Audio Engineering Society, 45(6), 1997, pp. 456-466.

[4] J. Garcia, T. Carpentier, J. Bresson, "Interactive-Compositional Authoring of Sound Spatialization," Journal of New Music Research, 46 (1), 2017.

[5] T. Carpentier, "A new implementation of Spat in Max," in Proceedings of the 15th Sound and Music Computing Conference (SMC), Limassol (Cyprus), 2018, pp. 184-191.

[6] J. C. Schacher, "Seven years of ICST Ambisonics Tools for MaxMSP - A brief report," in Proceedings of the 2nd International Symposium on Ambisonics and Spherical Acoustic, IRCAM, Paris, 2010.

[7] A. Sèdes, P. Guillot, and E. Paris, "The HOA library, review and prospects," in Proceedings of the 40th International Computer Music Conference and 11th Sound and Music Computing Conference, Athens (Greece), 2014, pp. 855-860.

[8] C. Miyama, G. Dipper, and L. Brümmer, "Zirkonium MKIII - A toolkit for spatial composition," Journal of the Japanese Society for Sonic Arts, 7(3), 2015, pp. 54–59.

[9] D. Ledoux et al., "SpatGRIS/ServerGRIS, Creative tools for 2D and 3D sound spatialization," in Proceedings of the 43rd International Computer Music Conference (ICMC), Daegu, 2018, pp. 291-297.

[10] J. Scordato, "Novel perspectives for graphic notation in IanniX," in L. Brümmer, S. Kanach, and P. Weibel (eds.), *From Xenakis's UPIC to Graphic Notation Today*. ZKM | Karlsruhe and Hatje Cantz, 2020, pp. 578-587.

[11] J. Scordato, M. Barberis, and N. Matteo, "Sketching Sonic Trajectories," in Proceedings of the Sound, Image and Interaction Design Symposium (SIIDS), Madeira, 2020.

[12] S. Tarakajian, D. Zicarelli, and J. Clayton. "Mira: Liveness in iPad Controllers for Max/MSP," in Proceedings of the International Conference on New Interfaces for Musical Expression (NIME), Daejeon, 2013, pp. 421-426.

[13] Messages and commands - IanniX Wiki - GitHub https://github.com/buzzinglight/IanniX/wiki/4.-Messages-and-commands#42-received-commands

REAL-TIME ALGORITHMIC TIMBRAL SPATIALISATION: COMPOSITIONAL APPROACHES AND TECHNIQUES

Stefano CATENA (stefano.catena23@gmail.com) [1]

[1] **Independent Researcher**, Milano, Italy

ABSTRACT

This paper introduces a series of techniques and approaches to algorithmic timbral spatialisation, the real-time processing of audio data and its musical organization in a binaural or multichannel listening space. The intent of the work is to explore different ways for the automated spatialisation of the audio spectrum, especially in contexts of electroacoustic and acousmatic music composition. Typical Digital Signal Processing operations and algorithms will be used in order to create and/or retrieve data for the dynamic positioning of the audio sources. Firstly, in order to describe the processes, the concept of timbral spatialisation will be introduced, describing the compositional interest of such approach. Then the different techniques for data generation will be formalized, describing their "audioparous" process, that is when information regarding musical organization is extrapolated from an audio source. The various approaches to interpretation and usage of the data will be discussed, as well as their implementation in SuperCollider.

1. INTRODUCTION

The concept of "spatiality" has been taken into consideration in the compositional practice since between the 10th and 14th century in the vocal music, particularly with the *antiphonal psalms* [1], later elaborated with the tradition of *polychoral music* [2]. Even during the classical period some works actively use the spatial component of sound ("Serenata 8 in D major for 4 orchestras, K. 286" from 1777 by W. Mozart), and later in the 19th and 20th centuries with composers such as Mahler ("Symphony No. 2") or Charles Ives ("Unanswered Question"). But it is only after the Second World War, with the introduction of electronic instruments and loudspeakers, that spatiality becomes a fundamental aspect of musical production, with some prominent composers, such as Karlheinz Stockhausen, who take great advantage from it [3]. In particular, since the 1970s, the spatial aspect has been one of the most in-depth and researched fields, [4], even with the design of special diffusion systems [5]. In recent years, with the greater accessibility and evolution of spatialisation techniques, several new concepts regarding spatiality have been introduced in the compositional practice. One can argue that spatial movement, even with little formal theory about it, has become a musical parameter just like rhythm, timbre and pitch [2]. One of the approaches to the organization of spatial parameter of sound is "timbral spatialisation" [6]. This process deconstructs a sound source into its individual spectral bands, addressing them as single point-sources, and placing them in the listening space. Timbral spatialisation then recombines the entire spectrum virtually, whether in the concert hall or in headphones, therefore not simply adding the spatial aspect at the end of the composition process, but actually re-composing the space [7]. Timbral spatialisation poses the problem of controlling each individual spectral band in space: such process can require potentially dozens of parameters, all at the same time. Some of the most used techniques for this task have been Wave Terrain Synthesis [8] or granular synthesis [9], but given the compositional nature of the process, the exploration of different algorithmic techniques for musical data generation is of relevant interest.

Sound and space are intertwined irreversibly: whenever sound is recorded, the space where the sound "happens" is recorded as well, inevitably being perceived with the spectral properties of the source [10]. A similar process of linkage can be used as framework for spatialisation, extrapolating data from audio sources in order to influence the spatial aspect of the music. Such technique can be called "audioparity", a composition mode in which musical data originate from sound. A compositional technique, consequently, is audioparous if it defines a projection between a source sound material and an outgoing musical organization [11]. For example, this compositional process has been used by composers such as Messiaen, as he integrated in his works transcriptions of singing birds [12]. With the usage of the same audio data for the spatialisation control, one can extend the concept of "audioparity" to "self-audioparity", where sound and space influence each other. We can therefore say that in "self-audioparous" techniques the musical organization is directly influenced and modified by the audio source itself.

In discussing algorithmic composition, even more when the concept of "audioparism" stands out, it is relevant to point out how human perception processes time scales differently. For example experiencing a simple sinusoid transposed to different time scale would change the perceptual results drastically, but the waveform itself would still remain the same [13]. While this is true for sonic experiences, it does not prevent the composer to cross the boundaries of time scales in order to "apply" data extracted

from one temporal region to another.

In the next chapters some of these techniques and approaches will be applied to real-time incoming audio signals in order create complex spatial textures within timbral spatialisation. In particular an exploration in the use of noise algorithms, the use of FFT descriptors and finally an audio snapshot and manipulation technique to automate the creation of auditory scenes.

2. APPROACH

While spatiality can be assimilated to other musical parameter such as pitch, rhythm and timbre, it is sometimes used as a simple post-processing effect, mostly because of the lack of a properly unified and defined spatial language [14]. The formalization of techniques for the generation of spatial data, provides a framework for spatialisation that can be easily integrated inside a composer's work. This is particulary true in electroacoustic and acousmatic composition where the space and spatial experience is aesthetically central [10]. Consequently, the spatial perception of the listener is of fundamental importance: how the movements are generated, how they affect spectromorphology of sound and how the spatialisation is connected, if it is at all, to its source. Even though there isn't a solid framework which might provide a reasonably secure basis for investigating space [10], it is possible to define spatial attributes and characteristics [14] that, in themselves, define and organize the spatial scene. These characteristics are inevitably crucial for the impact on the spatial experience of the listener, and must be taken into account when formalizing procedures for automatic spatialisation.

Timbral spatialisation enables the musical exploration of sound very differently from point-source techniques. It involves the "deconstruction" of sound into spectral bands, by means of several bandpass filters with different central frequencies, allowing for compositional processes to determine how the sound will be spatially distributed for each part of the spectrum. This process is similar to typical FFT synthesis and resynthesis, and even more to analog vocoders [15]. They allow for the deconstruction and reconstruction of the sound based on its frequency content; this process has been linked to the term *spectromorphology* [10]. The concept of "deconstruction" or "decomposition" of sound has been discussed previously [16], and other researches have examined concept and applications of timbral spatialisation as well [6, 17–19]. One can imagine that this particular approach is similar to the concept of orchestration, a term that acousmatic composers are very fond of [5]. They effectively "orchestrate" music on systems like the Acousmonium to achieve the desired spatial and sonic experience, creating and performing gestures based on their own personal taste but also on the spectromorphology of sound [20]. Similarly, timbral spatialisation is capable of creating diffused and immersive sound scenes [18], with the possibility of controlling each part of the sound spectrum algorithmically, creating new possibilities in spatial composition.

However, although the concept of splitting the sound into various frequency bands with bandpass filters is by itself straightforward, it poses several questions of compositional interest that may completely change the resulting spatial scene:

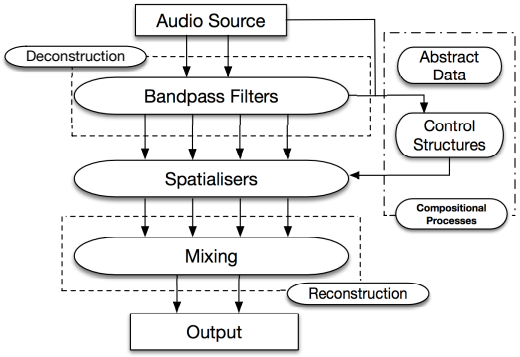


Figure 1. Flowchart of the timbral spatialisation process.

- How many frequency bands should be used?
- Which center frequency for each bandpass filter?
- Why? What's the effect on using different filter settings on the spatial experience?

For sake of consistency it is relevant that the process of deconstruction-reconstruction gives the most faithful result in comparison to the original: this means that is is desirable not to introduce any kind of distortion. In this specific case, different combinations of frequency bands (called Fs or frequency sets) were applied to the incoming signal, in order to experiment on various scenarios, as described in Table 1. In particular:

- Fs_1 frequencies are the preferred octave frequency bands according to the ISO standard [21];
- Fs_2 frequencies from the Random*Source - Serge Resonant Equalizer [1] ;
- Fs_3 frequencies represent a logarithmic scale;
- Fs_4 frequencies are from an API 560 graphic equalizer;

In each set, the lower and upper cutoff frequencies for a single bandpass filter are defined by a pair of values: e.g. [31, 63] are respectively the lower and upper cutoff frequencies for the first bandpass filter in Fs_4 (see Table 1).

The number of used frequency bands has a strong impact on the overall experience: Fs_3, that has only four spectral bands, is more focused and its particular spectral regions are highly localized, while the other frequency sets are more immersive and seem to be a more coherent group. However, different frequency sets can be used depending on the compositional goal and, possibly, on the incoming audio's spectromorphology. In terms of timbral spatialisation, anyway, the process may also be applied to narrower

[1] A unique ten-band filter designed specifically for electronic sound synthesis and processing, where each band, except for bottom and top two frequencies, are spaced at an interval of a major seventh.

Fs_1 (Hz)	Fs_2 (Hz)	Fs_3 (Hz)	Fs_4 (Hz)
22	29	10	31
44	61	100	63
88	115	1000	125
177	218	10000	250
335	411	20000	500
710	777		1000
1420	1500		2000
2840	2800		4000
5680	5200		8000
11360	11000		20000
20000	20000		

Table 1. The various frequency sets (Fs) used.

bands of frequency so that the full audio spectrum of the original source is not reproduced: this would yield non-contiguous but very localized spatial sound-shapes. In this case, the simplest scenario has been taken into consideration, which involves static center frequencies for the bands: further exploration of the spatial implications of this "dynamic" approach can be of relevant compositional interest.

3. TECHNIQUES - CONTROLLING THE SPATIOMORPHOLOGY

Nowadays, many spatialisation systems and techniques are still mixer-oriented [20]: this is because many masterpieces of acousmatic and electroacoustic music were composed for audio systems controlled by a mixer. This means that the interpretation of musical pieces has to be done manually: in the Acousmonium, for example, each fader in a mixer controls the volume of a speaker (or group of speakers) placed strategically in space, with its own frequency response. The history of electronic music interpretation regarding space, is strongly related with the sound-space aesthetic developed in these systems [19]. The development of several new technologies and techniques for controlling spatialisation has changed the way composers approach to this parameter, with very heterogeneous results [22]. The central point of spatialisation is, regardless of the technology used, the control of the spatial environment's attributes. Unfortunately these attributes are not definitive [14], not to mention the lack of a unified musical notation for spatiality and spatialisation [23]. This means that the control of the spatiomorphology of a composition is defined everytime by the composer, perhaps not even coherently with previous compositions. When music is performed by acoustic instruments in an acoustic environment, the physical level of description by itself often provides a workable roadmap to both the listener's experience and the composer's intent [14]: this isn't always true for acousmatic and electroacoustic music, where space is an aesthetically created "environment" [10].

For simplicity, the considered attributes are going to be the geometrical coordinates in a 2D plane (x and y, representing back, front, left, right positions) and a "distance-from-the-listener" (d) attribute: each of these parameters will be applied for every frequency band of the timbral spatialisation process. For example, in the Fs_3, at least twelve parameters (four frequency bands with three attributes each) will be necessary to manipulate the spatial scene. Moreover, the various frequency bands can be seen not just as single, individual point-sources, but also as a group or series of groups, reinforcing the spectral aspects of the spatialisation, with the possibility of "granular" [9] control over the bands. Picking x, y and d attritubes has been a subjective choice of the author. However, this choice provides a more general compositional framework: these attributes can be defined in any spatial environment (in various degrees), making it easier to switch between spatialisation technologies, or even between implementation languages.

As previously noted, audioparous techniques indicate a composition mode in which musical data originate from sound. We can formalize several audioparous procedures that can shape and define different aspects and timescales of the compositions, from micro to macro musical organization: e.g. single generative spatial gestures, the flocking movement of the whole timbral spatialisation etc.

However, differently from other examples of audioparous compositional procedures [11, 12], the application of such techniques in spatial contexts do not produce notation or sound per se, but rather modify and reshape an incoming audio signal. Furthermore, the actual incoming sound is an interesting parameter to explore in order to control sound itself: we can then extend the idea of audioparism to a "self-audioparism", where the incoming audio, perhaps through some other control process, spatialises itself.

3.1 Exploring noise

In the time domain, noise can be defined as sound in which the amplitude over time changes with a degree of randomness. The amplitude is maximally random in the so-called *white noise*. In the spectral or frequency domain, noise can be defined as sound that has a continuous power spectral density over a certain frequency bandwidth. The power spectral density of all frequencies is equal in white noise. [2]. Consequently, there are different "flavours" of noise, and they can be suitable for different compositional strategies.

Some synthesis techniques in analog electronic instruments from the "control-voltage" era are still alive and kicking: for example the use of sample and hold circuits to create random sequences by "feeding" them pink noise [24].

This particular techniques enables to sample a value upon a received trigger and hold it still until another trigger is received: if the sampled signal is noise, then the output would be a sequence of random value. Even more interesting is the interpolation between these values: instead of jumping quickly from one value to another, the transition is smooth, producing all the values in between as well, like in Figure 2. In a spatial context these values can be easily used to automate all the parameters needed for each frequency bands: it's simple to create a rich and immersive spatial scene by generating several of these functions. Furthermore, by controlling the sampling rate, the speed of

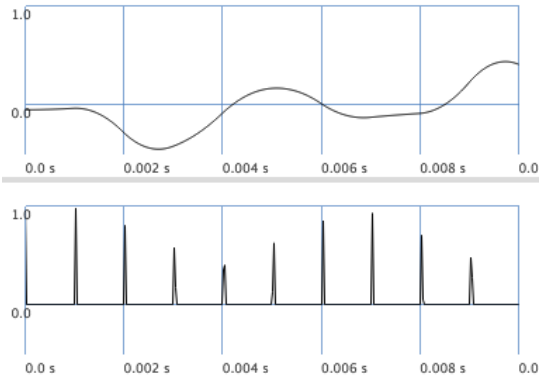


Figure 2. Quadratically interpolated random values sampled from a noise function with relative trigger inputs.

the change of spatial attributes would dramatically change: nesting noise functions into other noise functions to dynamically change the x and y positions of the frequency bands adds fluctuation and unpredictability (Figure 3).

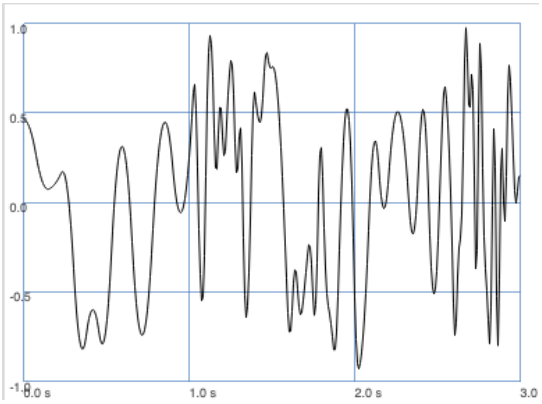


Figure 3. A sampled noise function with its sampling rate modulated by another nested noise function.

3.2 FFT controlled walk

The Fast Fourier Transform (FFT) inspects the frequency content of a signal, and can be extremely useful for audio analysis or frequency-domain sound processing. By "windowing" a real-time signal, a succession of overlapped spectral frames are obtained: the FFT focuses attention on the magnitude and phases values for all of the different frequency bin resulting from this operation [15]. By using this data, it is possible to spectrally analyze sound, extrapolating some of its intrinsic characteristics: these attributes can be used compositionally in a spatial context by mapping them to the relevant values, in our case the x, y and d attributes described in Section 2.

In this context, the so-called *Instantaneous Spectral Descriptors* [25] are used. They are a set of instantaneous attributes obtained from the FFT analysis that describe the spectral shape of sound in a certain moment in time: we could say that they are a photograph of the spectromorphology of audio. In particular, three descriptors were used:

- *Spectral Centroid*: the weighted mean frequency, or the "centre of mass" of the spectrum. It can be a useful indicator of the perceptual "brightness" of an audio signal;
- *Spectrum Roll-off point*: the frequency below which lies the 90/95 percent of the signal energy. This somewhat indicates the harmonic/noise cutting frequency;
- *Spectral Flatness*: it measures of "noisiness" or "sinusoidality" of the spectrum (or parts of it);

These descriptors are used as dynamic controllers for random walks, also called Brownian motion [26]. Specifically, two random walk functions generate the x and y position for each of the frequency bands: respectively, the Spectral Flatness and Spectral Centroid descriptors control the frequency and step's amplitude for the previously defined functions. This means that depending on measures of "noisiness" and the perceptual "brightness" of the incoming audio, there will be a changing number of steps per second and each of them will move closer or further away from the previous one. The Spectrum Roll-off point descriptor, instead, will control the distance attribute for the whole group of frequency bands acting as a global modifier and effectively treating them as a collective group that behaves coherently. The use of FFT analysis in this particular compositional technique, defines what was previously discussed as "self-audioparism": using the spectral descriptors values obtained by the analysis of the incoming audio onto itself, we define a self-sustaining, ever-changing algorithmic spatial texture.

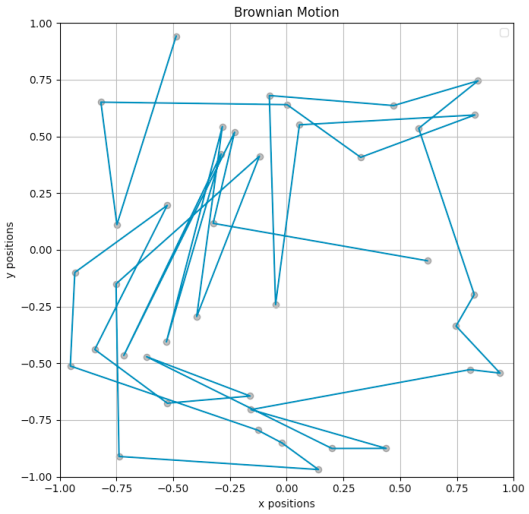


Figure 4. A 40 steps walk for a single frequency band in the 2D space, where (0,0) is the listener's virtual position.

3.3 Audio snapshot technique

The audio snapshot is yet another audioparous technique that involves sampling into a buffer a rather small snippet of sound, using this data to drive the spatialisation. This techniques is actually self-audioparous, because the sound being recorded is indeed from the source itself. The basic idea is to use an interpolating buffer player set to very low playback rates in order to read data: in this way we "transpose" information from a *micro* timescale (a few milliseconds of audio) to a *sound object* or *meso* time scale (from several tenths of a second to a few seconds of audio) [13] .

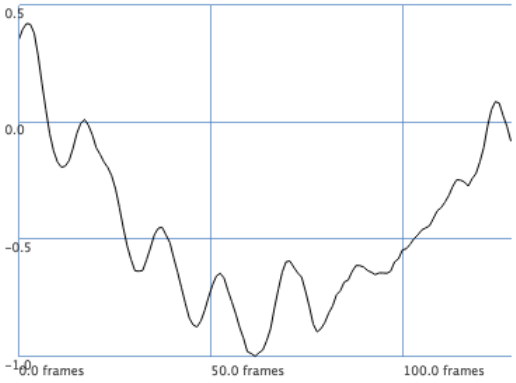


Figure 5. One snapshot of 128 frames, taken from *Solar Ellipse* by Barry Truax.

This particular technique of data generation, in practice, produces pseudo-random spatial gestures, depending on the length of the buffer: from 16 to 64 frames the gesture is definitely short, while from 128 onwards the spatial phrase starts being long enough to reach into the *meso* timescale. This, of course, depends on the playback rate of the buffer players: extending its frequency into audio territory (over 20Hz), one could even have audio-rate spatial modulation.

More precisely, for each of the frequency band, three buffer players will read the data: one for x position, one for y position and one for the distance attribute. Each of these players will read the buffer from random initial positions in order to get coherent but varied results: the combination of these three functions in time will be the actual gestural output applied to the spatial texture. Randomizing both the playback rate and the initial reading position, will also ensure that the resulting spatial gesture is always different for each of the spectral bands, giving the impression of a richer spatial scene. At any time the initial buffer can be resampled, loading new information and starting back from scratch with new audio data, producing completely new spatial motions.

Furthermore, once the buffer has been filled, typical Digital Signal Processing operation can be applied to the recorded data: for example, using sub-audio or audio generators with appropriate parameters will yield interesting and extreme results (see Figure 6).

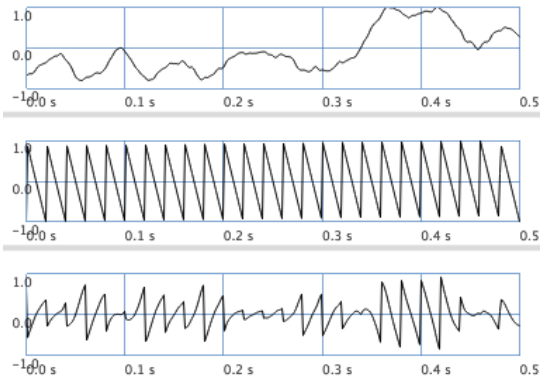


Figure 6. In the upper graph is the original sampled signal; in the middle graph 50Hz sawtooth wave acting as a modulator; in the lower graph the resulting signal from the multiplication of the two previous signals.

4. IMPLEMENTATION

The implementation of both the timbral spatialisation and the control techniques poses some challenge in the organization of the data flow, and can be relatively CPU heavy due to the spatial rendering of many frequency bands: in experimenting with these techniques, an Ambisonics [27] binaural approach was used. However, other technologies (such as DBAP [28], for example) may be more suitable for multichannel setups, both from perceptual aspect (no sweetspot) and computational aspects (much cheaper).

All the software was implemented in the SuperCollider environment [29] which features an Object Orientend programming language that controls a powerful audio synthesis server. The use of an audio dedicated programming language is particularly fitting because it comes pre-loaded with algorithms and Unit Generators that can be easily integrated in the workflow. Moreover, the SuperCollider community provides a series of free classes and add-ons [2] that dramatically extend the capabilities of both the language and the synthesis server. One of these free plugins has been used to implement the binaural spatial rendering, the "Ambisonics Toolkit [3] " (Atk), and more precisely the FOA ("First Order Ambisonics"). With a process of encoding, transforming and decoding (Figure 7), the Atk effectively allows the user to easily spatialize sound by controlling the transformation procedures, and specifying the nature of encoder and decoder(s). In this case, the spatial

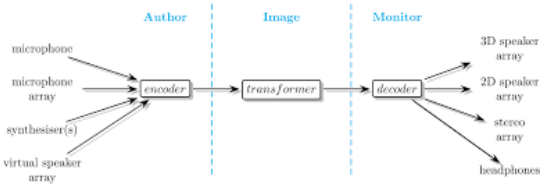


Figure 7. The Ambisonics workflow.

[2] https://github.com/supercollider/sc3-plugins
[3] https://www.ambisonictoolkit.net/

rendering is obtained through a binaural decoder based on the IRCAM's Listen HRTF database [4] which provides various equalization for different head's widths, allowing the user to choose their preferred settings. The spatial render is interchangeable with any appropriate binaural render, providing several of this decoders with different properties: for example a Synthetic spherical head model HRTF [30] or a CIPIC HRTF database from the University of California Davis [5].

It is important to mention that the SuperCollider architecture is based on the dualism between language (*sclang*) and server (*scsynth*) [31]: many processes can be implemented within both the interpreted language or the synthesis server, depending on the user's need. Specifically, the techniques and approaches presented here, have been implemented server side: in other words most of the functions such as noise generators, oscillators, FFT analysis etc.. are specific Unit Generators that are allocated and managed dynamically on *scsynth*. The advantage of this organization is that, other than the readiness of use, the low-level (C++) implementation of the Unit Generators allows for a much more optimized use of the computational power. Furthermore, *scsynth* offers a flexible and multichannel bus system which is perfect for sending/receiving the large number of computed data in the spatialisation system.

In SuperCollider there is a great choice of Ugens that can be used inside this compositional framework, or even substitute the ones presented here. For example:

- a number of "coloured" noise Ugens or stochastic generators such as WhiteNoise, PinkNoise, BrownNoise, GreyNoise or Crackle (a noise generator based on a chaotic function);
- various degrees of interpolated or non-interpolated sample and hold Ugens such as LFNoise0 (non interpolated), LFNoise1 (linearly interpolated) or LFNoise2 (quadratically interpolated);
- different types of Ugens suitable for real-time audio analysis such as Pitch (autocorrelation pitch follower), Amplitude (envelope follower), Loudness (extraction of instantaneous loudness in sones) etc..
- several add-ons are present, ranging from the sc3-plugins to a large suite of audio analysis tools called "Fluid Decomposition Toolbox" [32], all freely available;

While for each spatialised frequency band the *x* and *y* positions are straightforward (representing front, back, left and right), the *d* attribute described in Section 3 has been implemented according to [33] on distance cues. In order to emulate these cues, the *d* parameter scales the amplitude of each frequency band so that the direct signal decreases in amplitude more with distance than does the reverberant signal.

[4] http://recherche.ircam.fr/equipes/salles/listen/

[5] http://interface.cipic.ucdavis.edu/sound/hrtf.html

The implemented techinques are available publicly on the author's GitHub [6].

5. CONCLUSIONS

Timbral spatialisation is a signal processing technique that has a great potential for creating rich and fascinating spatial textures, but it can also be viewed as a tool for composing space and effectively considering it a part of the compositional workflow. Together with algorithmic techniques for the control of the spatial environment, it is possible to automate the set of attributes that define such virtual space. Furthermore, the introduced concept of "self-audioparity" adds another layer of complexity and coherence to the whole spatialisation process.

The possible applications for the discussed techniques are multiple and in different contexts:

- live spatialisation of electroacoustic and acousmatic performances;
- the reinterpretation in multichannel setups or binaural rendering of fixed media compositions;
- as a standalone tool for spatial composition and for the integration of space inside of a composer's workflow;

It is important to notice that the implemented techniques are described from a compositional point of view, which means that the final user can adjust the internal parameters and mappings according to its own taste: the specifications collected here are a representation of what is possibile, but are by no means definitive. Moreover, many other techniques can be implemented, or perhaps expanding the ones that have been presented: for example, further explorations of different spectral descriptors and its mapping to spatial attributes; the implementation of "dynamic" instead of "static" frequency bands; the creation of completely new audioparous algorithms from the ground up.

The next steps will involve the development of a framework for easily switching between techniques in real-time; inclusion of the "height" attribute in the spatialisation process; the implementation of a GUI for visual feedback; the integration of machine learning techniques for intelligent spatialisation; the creation of a set of hardware tools for the performative control of the spatial textures.

6. REFERENCES

[1] W. J. Solomon, "Spatialisation in music: the analysis and interpretation of spatial gestures," Ph.D. dissertation, University of Georgia, 2002.

[2] C. Roads, *Composing electronic music: a new aesthetic*. Oxford Univerisity Press, 2014.

[3] K. Stockhausen, "Music in space," *Die Reihe*, vol. 5, pp. 67–82, 1961.

[6] https://github.com/StefanoCatena/Timbral-Spatialization/tree/main

[4] F. Schumacher and C. Fuentes, "Space–emotion in acousmatic music," *Organised Sound*, vol. 22, pp. 394–405, 12 2017.

[5] S. Desantos, C. Roads, and F. Bayle, *Acousmatic morphology: an interview with Francois Bayle*. Computer Music Journal, Fall 1997, vol. 21, no. 3, pp. 11–19.

[6] R. Normandeau, "Timbre spatialisation: the medium is the space," *Organised Sound*, vol. 14, pp. 277 – 285, 12 2009.

[7] S. James and C. Hope, "2d and 3d timbral spatialisation: Spatial motion, immersiveness, and notions of space," *International Computer Music Conference*, pp. 77–84, 01 2013.

[8] S. James, "Spectromorphology and spatiomorphology of sound shapes: Audio-rate aep and dbap panning of spectra," *International Computer Music Conference*, pp. 278–285, 09-10 2015.

[9] D. Kim-Boyle, "Spectral and granular spatialization with boids," *Computer Music Journal*, pp. 139–142, 01 2006.

[10] D. Smalley, "Space-form and the acousmatic image," *Organised Sound, 12(1)*, 2007.

[11] A. Valle, "Sampcomp: sample-based techniques for algorithmic composition," in *Proceedings of the 22nd CIM*, November 20-23 2018, pp. 128–135.

[12] O. Messiaen, *The Techniques of my Musical Language*. Paris: Leduc, 1956.

[13] C. Roads, *Microsound*. MIT Press, 2001.

[14] G. Kendall and M. Ardila, "The artistic play of spatial organization: Spatial attributes, scene analysis and auditory spatial schemata," *Computer Music Modeling and Retrieval. Sens of Sounds, 4th International Symposium, CMMR*, pp. 125–138, 08 2007.

[15] M. Dolson, "The phase vocoder: A tutorial," *Computer Music Journal*, vol. 10, pp. 14–27, 12 1986.

[16] K. Stockhausen, *Vier Kriterien der Elektronischen Musik*. DuMont Buchverlag, 1970- 1977, vol. 4, pp. 360–401.

[17] R. Torchia and C. Lippe, "Techniques for multi-channel real-time spatial distribution using frequency-domain processing." *Conference on New Interfaces for Musical Expression (NIME04)*, pp. 116–119, 06 2004.

[18] S. James, "A classification of multi-point spectral sound shapes," *Proceedings of the ACMC/AFAE Conference*, pp. 57–64, 07 2016.

[19] A. Barberis, S. Mungianu, and S. Sapir, "Ormé, a tool for automated spatialization of fixed-media music based on spectrum contents annotation," *Colloqui d'Informatica Musicale (CIM)*, 09 2016.

[20] A. Vande Gorne, *L'interprétation spatiale. Essai de formalisation méthodologique*. Université de Lille, December 2002.

[21] "Acoustics, Preferred frequencies for measurement," ISO 266:1997(E), 1997.

[22] N. Peters, G. Marentakis, and S. Mcadams, "Current technologies and compositional practices for spatialization: A qualitative and quantitative analysis," *Computer Music Journal*, vol. 35, pp. 10–27, 03 2011.

[23] A. Vidolin, "Suonare lo spazio elettroacustico," *Quaderni di Musica*, vol. 51, 2002.

[24] A. Strange, *Electronic Music: Systems, Techniques and Controls.* Dubuque, Iowa: William C. Brown Company, 2nd Ed., 1983.

[25] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the cuidado project," Institute for Researche and Coordination in Acoustics/Music, Tech. Rep., 01 2004.

[26] A. Valle, "Algorithmic spatialisation. interfacing supercollider to the wavefield system," CIRMA/DAMS, Department of Humanities (StudiUm), University of Torino., Tech. Rep., 2016.

[27] R. K. Furness, "Ambisonics-an overview," in *Audio Engineering Society Conference: 8th International Conference: The Sound of Audio*, May 1990.

[28] T. Lossius and P. Baltazar, "Dbap – distance-based amplitude panning," *International Computer Music Conference*, 01 2009.

[29] S. Wilson, D. Cottle, and N. Collis, *The SuperCollider Book*. Cambridge, Mass.: The MIT Press, 2011.

[30] R. O. Duda, "Modeling head related transfer functions," *Proceedings of the Twenty-Seventh Annual Asilomar Conference on Signals, Systems and Computers*, 1993.

[31] A. Valle, *Introduction to SuperCollider*. Logos Verlag, 2015.

[32] P. Tremblay, O. Green, G. Roma, and A. Harker, "From collections to corpora: Exploring sounds through fluid decomposition," *International Computer Music Conference and New York City Electroacoustic Music Festival - ICMC-NYCEMF*, 2019.

[33] J. Chowning, "The simulation of moving sound sources," *Computer Music Journal*, vol. 1, 06 1977.

IVES - Interactive Virtual Environment System: A modular toolkit for 3D audiovisual composition in Max

Damian Dziwis (damian.dziwis@th-koeln.de) [1,2], **Johannes M. Arend**[1,2], **Tim Lübeck**[1,2], and **Christoph Pörschmann**[1]

[1]**TH Köln - University of Applied Sciences**, Cologne, Germany
[2]**Technical University of Berlin**, Berlin, Germany

ABSTRACT

The *Interactive Virtual Environment System* (IVES) is a toolkit aiding the production of immersive audiovisual 3D virtual environments for screen-based or virtual reality (VR) applications with loudspeaker- or headphone-based spatial audio reproduction. It is developed within Cycling´74s Max programming environment and consists of a set of interface-based, higher-level building-block modules, similar to the BEAP and VIZZIE toolkits included in Max. IVES uses and unifies established programming libraries such as Jitter / OpenGL (Cycling'74), Spat (IRCAM), VR (Graham Wakefield) into ready-to-use abstractions with graphical user interfaces (GUIs). This allows simple patching of individual spatial audio and visual 3D rendering chains. IVES provides various blocks in a flexible modular patching system, suitable for audiovisual rendering in different application scenarios with different content, and manages the synchronization and conversion of data, control messages and coordinate-systems, used differently in the underlying libraries. Furthermore, the system also provides modules for the creation, interaction, motion, and transformation of audiovisual spatial elements within virtual environments. The toolkit allows users to concentrate on the composition and artistic content in audiovisual virtual environments rather than the programming of the complex systems behind them.

1. INTRODUCTION

In the fourth movement of his composition "Symphony No. 4" [1], Charles Ives uses a percussion ensemble spatially separated from the orchestra as a form of spatial composition technique. This type of, in this context, unconventional placing of sounds in space is only one of countless examples, representative of the interest of composers in the use of spatial features in their works. Placing sounds in space, like Ives did in the concert hall, moving them around and using the room acoustics and their perception, are common techniques used in spatial composition. In the context of electroacoustic/acousmatic music and spatial audio reproduction systems with multi-channel loudspeakers or headphone-based binaural rendering, these techniques have been further developed and also extended. Besides more complex options like the movement with three-dimensional trajectories, also new techniques such as spatial sound processing and the simulation of different room acoustics became possible [1]. This led to the emergence of various software aiding the spatial composition process (e.g., HoloEdit [2, 3], Zirkonium [4, 5]) and spatial audio reproduction (e.g., Spat [6, 7], ICST Ambisonics Tools [8, 9], AudioScape [10]) in common computer music programming environments (e.g., Max [11], PureData [12], SuperCollider [13]).

[1] Symphony No. 4, S. 4 (K. 1A4), 1925; Charles Ives (1874–1954)

1.1 Audiovisual Virtual Environments

Spatial audio plays an important role not only in music but also in other media such as movies and games. In interdisciplinary art, the interest of media artists and composers in combining these media to create spatial audiovisual works also increased. Especially the recent developments and availability of virtual and augmented reality (XR) systems led to new possibilities to explore audiovisual spatial environments for art production.
Emerging from the gaming industry, game engines (such as Unity or Unreal) represent powerful tools for creating and rendering these spatial audiovisual environments for screen- and XR-based applications, even beyond their primary application for games. While also being frequently used for audiovisual spatial composition, game engines can still have shortcomings in several application scenarios. Being very powerful especially in the visual domain, the possibilities in audio might be too limited for many composition requirements. Real-time multi-channel input, sound synthesis, and generative composition procedures can be challenging in game engines. Also, deeper control over the audio processing and the synthesis of spatial sound fields may be desirable for many applications. This has led to several developments that combine game engines and their extensive capabilities in 3D visual environments with computer music languages for composition, sound synthesis, and audio spatialization [14–16].

1.2 Audiovisual Rendering in the Max Environment

While game engines with their graphical editors and integrated physics and rendering engines are attractive tools for creating visual 3D worlds, their basic functionality can

also be programmed in Max. Max is a graphical programming environment, originally focused on audio and music programming and often used by computer music composers and media artists. The Max library Jitter for visual components has in addition to objects for graphics and video also an implementation of OpenGL for programming and rendering 3D environments.
The integration of virtual reality (VR) systems is realized with the VR library for Max [17]. The library allows implementing head-mounted-display (HMD) reproduction, tracking devices, and controller integration.
These libraries make the programming of virtual environments and their rendering within Max a potential alternative to the use of game engines. In combination with a spatial audio and composing library, such as the well-established Spat library by IRCAM, Max users can program their custom virtual environment systems and rendering pipelines using these libraries. This gives them deep control over the underlying processes in a programming environment dedicated to audiovisual composition.
Still, even with the help of these libraries, programming such systems can be very challenging. The implementation requires broad knowledge about spatial audio reproduction, 3D virtual environments, as well as rendering pipelines and their parametrization. Furthermore, exchanged data must be converted and adjusted, for example, because of different coordinate conventions used by the libraries.
Toolkits such as COSM [18] aid the programming of virtual worlds and rendering pipelines by offering high-level programming externals. It provides objects for spatial visual and audio rendering, as well as the creation of virtual worlds. Nevertheless, also with high-level externals, the process of creating audiovisual virtual environments is still quite programming intense.

1.3 The IVES Toolkit

In an attempt to minimize the programming effort required to realize audiovisual virtual environments within Max, we present in this paper the development of "IVES - Interactive Virtual Environment System", a modular toolkit for 3D audiovisual composition in Max. IVES is a set of higher-level building-block modules with graphical user interfaces (GUIs) aiding the patching of virtual environments and rendering pipelines. Within the toolkit, we use the Spat library to program the spatial audio rendering and composition modules, Jitter with OpenGL for visual 3D environments modules, and the Max VR package to integrate HMD-based VR systems. The conversion of data and control messages used in between these libraries is done inside the modules, ensuring consistent use of parameters and coordinate conventions.
As a result, the toolkit provides ready-to-use abstractions with interfaces for parametrization with no additional programming required, while it fully integrates in the Max development environment offering every kind of programmable extension. The modular system preserves a deep control over the signal and rendering processing and allows the adaptation to different source data and application scenarios. This implies the creation of a 3D world and simulated sound field with virtual sound sources and corresponding visual elements, or the use of recorded or real-time microphone array input in combination with audio-reactive 3D visuals. Composers and media artists can create virtual environments presented on VR systems with six-degrees-of-freedom (6DoF) tracking and dynamic binauralization over headphones, as well as works for a concert space with projection and loudspeaker-based sound reproduction. They can integrate various controllers, interfaces, and tracking-systems available in Max and program their own generative systems, algorithmic compositions, or sound synthesis used as content in their audiovisual environments.
The IVES toolkit aims to guide the creation of 3D virtual environments with a focus on audio and music. Unlike game engines, which are very powerful in the visual domain, IVES is used to embed a 3D virtual environment engine in an audio and music specific programming language such as Max, which is quite common among composers and media artists. It provides an easy-to-use solution for artists, especially those already familiar with Max or similar visual languages, and gives them deeper control and more options for real-time audio, spatial composition, sound and their spatialization compared to game engines, with less complexity than combined approaches. Thus, the toolkit allows artists to use the basic features of visual 3D environments, VR, and sound spatialization without having to learn a new environment or language, or programming of the principles behind rendering and spatialization.

2. DESIGN AND IMPLEMENTATION

The IVES toolkit's technical design follows the idea of a modular system. It consists of higher-level GUI-based blocks, as introduced with the BEAP and VIZZIE toolkits, which are integrated in the Max programming environment. This modularity allows the user to build up individual signal-processing chains by connecting modules with patch cords. The visual programming paradigm used in Max, with programming instructions as objects/nodes that are also connected to each other by patch cords, is very similar to this principle. This facilitates the adaptation and implementation of such a modular system paradigm. While the Max programming language can be considered to be already very high-level, the main difference of the here presented blocks is that they do not require further lower-level programming of functionality. Instead, they represent modules that can be connected in the order of the required processing chain and parameterized over their GUI. In its current development state, the available modules in IVES can be divided into 3 categories:

- Spatial audio modules to create and process a spatial sound field. Those modules provide the required rendering chain for loudspeaker- or headphone-based reproduction. Furthermore, the modules allow to transform and interact with the sound field in the spatial audio domain.

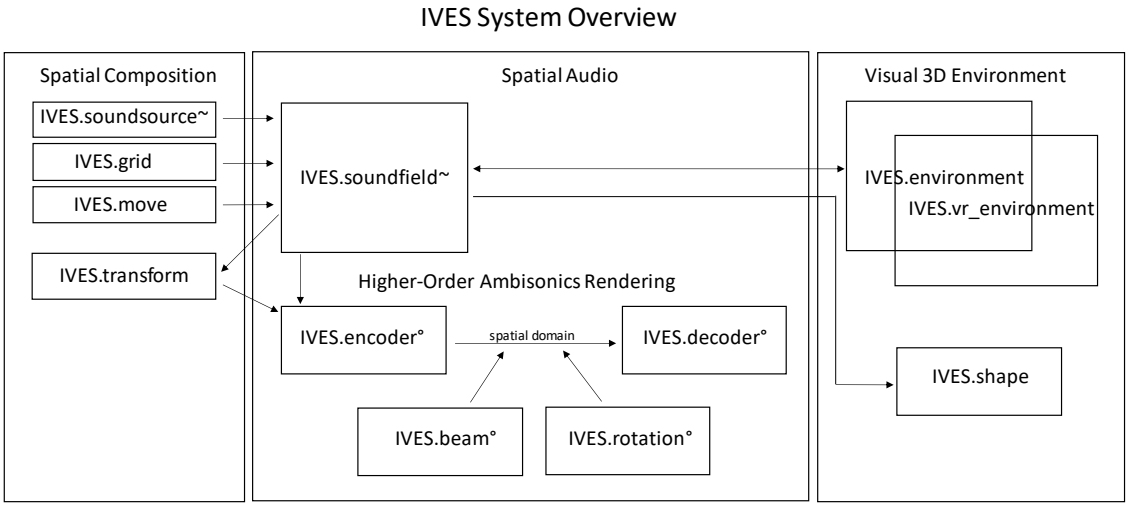


Figure 1. IVES system overview. The diagram shows the architecture with the spatial audio rendering as the center of the toolkit. It shows the connections of spatial composition, as well as visual 3D environment modules to the spatial audio rendering.

- Visual 3D environment modules that set up a virtual world with the needed rendering chain implemented. The modules allow to create 3D objects in the virtual environment and link them to virtual sound sources.
- Control and interaction modules, for example, to generate grids/patterns of object positions and trajectory movements, transform object parameters, and simulate/control head rotation parameters.

The system architecture of IVES (see Fig. 1) is centered around the virtual sound field. This indicates the slightly stronger focus on audio in IVES, as opposed to the game engines discussed. The visual environment is linked to the data of the virtual sound field. Virtual sound sources are automatically represented as visual objects in the virtual environment and can be linked to added 3D objects. For example, parameters such as the listener's position and head orientation are synchronized as position and view of the camera in the virtual environment. Still, the visual environment, as well as the spatial sound field, can be used and parameterized independently and self-sufficient.

In the current development state, the externals of three libraries are used to program the modules for creating, interacting, and rendering audiovisual virtual environments in Max.
The OpenGL [19] objects of the Jitter library by Cycling'74, integrated in the Max 8 (v.8.1.10) programming environment, are used to render a basic visual virtual environment and place 3D elements in it. The VR library by Graham Wakefield (v.1.0.1) is used to integrate PC-VR systems, compatible with SteamVR, Oculus, or Vive drivers, and render the virtual environment on HMDs and use tracking for 6DoF rendering of the virtual environment.
The sound spatialization is implemented using the Spat 5 (v.5.2.1) library by IRCAM. It is used to set up the virtual sound field and an Ambisonics-based [20] processing chain to manipulate and render the sound field for loudspeaker-based or binaural headphone reproduction. Whereas Max 8, as well as all implemented libraries, are compatible with Microsofts Windows and Apples MacOS platforms, the VR functionality has only been tested on Windows 10 operating systems.

2.1 Spatial Audio with Spat 5

The Spat library by IRCAM is a sophisticated library providing many externals for sound spatialization and spatial composing. It can be used for many different spatialization scenarios, such as reproduction on loudspeaker- or headphone-based systems. It offers all necessary functions and algorithms for spatial en- and decoding, reverberation, sound processing, plotting and visualization, as well as tools for motion and trajectories of sound sources. Even though the Spat library is already providing an all-in-one sound spatializer (*spat5.spat~*) with a graphical interface, reverberation, and a fully integrated rendering chain, it can be useful for many situations to implement separated or different processing chains. As all elementary spatialization algorithms and functions are also provided as separated externals, one of the main purposes of the Spat library is to enable the programming of individual spatializers, tailored to different application scenarios (see, for example, different Spat-based spatializers as Max4Live devices [21, 22]).
IVES is also built on a custom implementation to achieve the flexibility of a modular system and to allow the reproduction of simulated sound fields or multi-channel audio from spherical microphone arrays. It allows transforming the sound field, as well as the reproduction on various multi-channel loudspeaker systems or headphones. IVES uses the Spat externals with the MC multi-channel system

introduced in Max 8, providing bundled multiple signal chords to simplify patching and connection of the different audio modules. The number of channels is automatically adapted by IVES modules to the used input channels or the spatial sound field order.

2.1.1 Sound Fields in IVES

IVES.soundfield∼ represents one of the core modules of the toolkit and can operate in two different modes:

- Virtual Sound Sources:
  This mode handles spatial parameters of the listener as well as virtual sound sources and their signal processing. It uses the *spat5.viewer* object to manage the position of sound sources and the listener in the virtual space. Additionally, the listener orientation is considered to enable full 6DoF roaming with dynamic binauralization in the appended rendering process. Sound sources can also be freely placed in the sound field coordinate system. To enable distance perception of the virtual sound sources, gain attenuation is implemented and applied to the source signal. The distance between the listener and each source is calculated, resulting in a gain factor based on the inverse-distance law.
  In addition to the distance attenuation, the source pre-processing provided by the Spat library's *spat5.source∼* object is implemented to generate a continuous pre-delay based on distance to produce a Doppler effect. The processed signals of the sources, as well as the spatial parameters of all elements in the coordinate space, are subsequently passed to the spatial encoder.

- Multi-Channel Audio:
  The IVES toolkit also supports rendering of sound fields recorded with spherical microphone arrays. Thus, this mode of the sound field module aids the playback of multichannel audio files containing such recordings. It is dynamically generating MC signal streams according to the corresponding channel number. Alternatively, for example for real-time input, the input stream can directly be connected to the spatial encoder.

2.1.2 Higher-Order Ambisonics En- and Decoding

The spatial audio rendering is performed with a Higher-Order Ambisonics (HOA) processing chain (see Fig. 2). It consists of an encoder and decoder module that can be applied to the described above modes and used for different reproduction scenarios.
HOA represents the Ambisonics 3D-audio format in spatial resolution orders greater than one. Ambisonics is an object-based full-spherical spatial sound format, that, unlike channel-based systems, is agnostic to channel numbers or speaker positions. While higher numbers of channels and loudspeakers lead to higher orders and resulting spatial resolution, the decoding can be adopted to various reproduction scenarios. The HOA spatial format was chosen as it can encode single virtual sound sources as well as microphone array recordings. The encoding results in the B-Format sound field representation, offering high flexibility of decoding to various loudspeaker layouts as well as binaural headphone reproduction. The Spat library offers HOA encoders for simulated sound fields and several common microphone arrays. Decoders are available for loudspeaker reproduction or as binauralizer for headphones.
The encoder module (*IVES.encoder°*), includes several HOA encoders provided by Spat 5. One HOA encoder for simulated sound fields and three encoders for microphone arrays. The Zylia ZM1, Eigenmike EM32, and 1st order A-Format microphones, such as the Sennheiser Ambeo VR mic, are supported. The encoders can be selected and parameterized with the module's GUI. Parameters required through the whole rendering chain, such as spatial order, dimension, or normalization, are passed automatically to subsequent modules.
The decoder module (*IVES.decoder°*), includes a HOA decoder for multi-channel loudspeaker reproduction as well as a HOA binauralizer for spatial headphone reproduction using head-related transfer functions (HRTFs).
For the loudspeaker decoder, a grid generator was implemented. It can generate loudspeaker positions for common grids and loudspeaker layouts as provided by the Spat library. To enable dynamic binauralization also for recorded sound fields with microphone arrays, head rotation parameters need also to be considered in the decoding process.
Because the HOA binauralizer provided in Spat 5 does not support parameters of head rotation for dynamic binauralization, a sufficient sound field rotation needs to be implemented. For this, IVES rotates the sound field around a static listener's head position (in contrast to a rotation from the listener's point-of-view). As the HOA rotation implemented in Spat 5 is by default intrinsic (following the 'ZYX' convention), the rotation parameters are converted to an extrinsic rotation order (following the 'XYZ' convention). This ensures an adequate sound field rotation for a listener's point-of-view with given Euler angles (yaw/pitch/roll) as head orientation parameters.

2.1.3 Spatial Domain Manipulation

The IVES toolkit also provides sound field manipulation modules operating directly on the spatial sound field. This sound field interaction, transformation, and manipulation are done in between the en- and decoding spatial processing, in the so-called spatial domain (also called Ambisonics or spherical harmonics domain). In its current development state, a rotation and beam-former module for transformations in the spatial domain are implemented in IVES. The rotation module provides a spherical Euler rotation of the given sound field around the listener. The *IVES.beam°* module provides an implementation based on the *spat5.hoa.focus* external, allowing a selective fragmentation of the sound field based on orientation and selectivity of the beam. It is based on the concept of beamforming, a spatial filtering technique that can be used to filter spatial

Figure 2. Spatial audio pipeline using IVES. The example shows a simple simulated sound field with one virtual sound source. It is based on 3rd order HOA rendering with binaural reproduction. It consists of a virtual sound field module with a single sound source module as audio input and the head rotation module for 3DoF motion tracking. The audio stream and position data from the sound field module is then spatialized using the en- and decoder modules.

sound with a given directivity.
Both modules can be parameterized in the GUI and also linked to external interfaces and controllers, which allows designing new forms of interactions and spatial compositions [23]. In contrast to spatial composition techniques such as the positioning and movement of single sounds in a simulated sound field, the techniques based on the interaction and transformation in the spatial domain can also be applied to recorded sound fields with microphone arrays.

2.2 Visual 3D Environments with Jitter and VR

The processing chain for visual 3D environments is based on Max Jitter and its implementation of OpenGL. Jitter is Max's integrated library for image, video and 3D graphics processing. Similar to the MIDI and audio processing objects with Max and the MSP library, Jitter offers objects for programming signal processing with visual elements. This includes 3D graphics using OpenGL, an open-source application programming interface (API) for 2D and 3D graphics rendering. To enable PC-VR systems, including the presentation on HMDs and integration of tracking data for 6DoF movement, the VR package for Max was integrated. VR is a third-party library made by Graham Wakefield and the Alice Lab for Computational Worldmaking at York University to integrate VR hardware and stereoscopic rendering on HMDs into the Max programming environment.
The core module for handling visual environments in IVES is *IVES.environment*. It is the visual equivalent to *IVES.soundfield~*. When connected to the *IVES.soundfield~* module, it synchronizes with the elements in the virtual sound field (number and position of sound sources, listener position and orientation for the

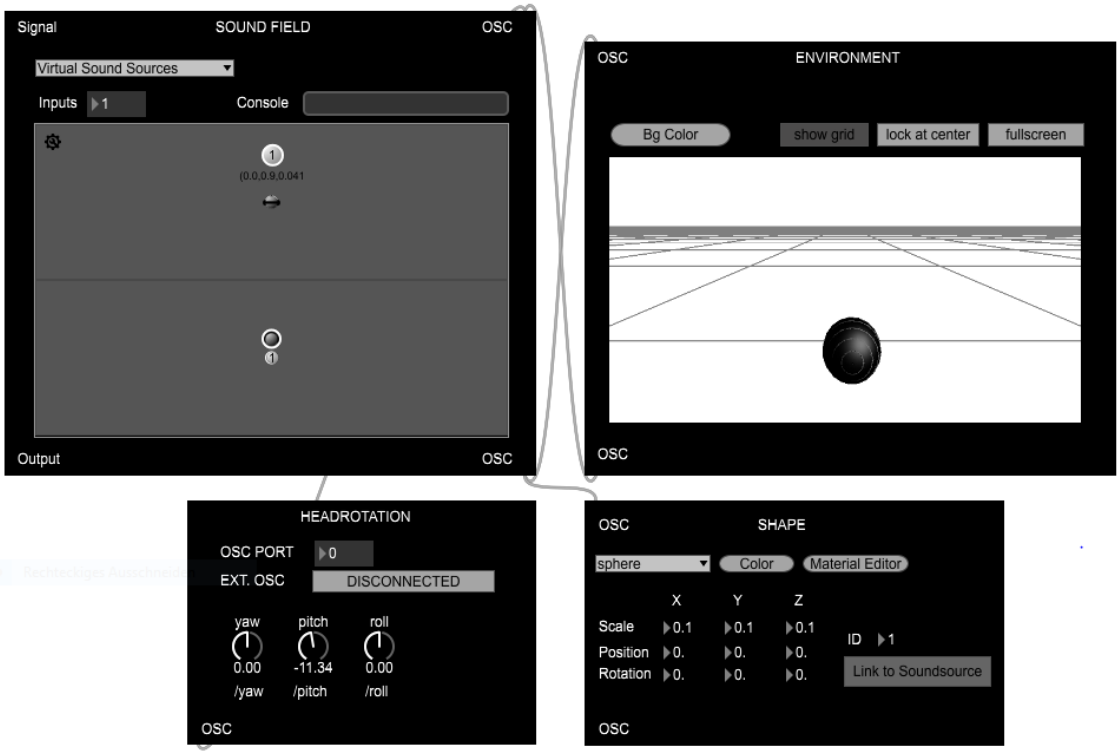


Figure 3. Visual 3D environment in IVES. The example shows a simple 3D environment with one sphere object linked to a virtual sound source. It consists of the sound field module as the core of the virtual environment. The environment module creates a visual representation of the sound field, a shape module renders a black sphere at the position of the given virtual sound source. The head rotation module is connected to the sound field and used to enable 3DoF motion in the audiovisual virtual environment.

viewer camera). The main difference to *IVES.soundfield~* is the implementation of the rendering chain inside the module. A basic environment with a world and physics context is set up, and the rendering of an empty world showing a grid. When sound sources are added to the sound field, they automatically show up as visual objects in the environment. The *IVES.shape* module allows to add further 3D objects. The module offers basic parametrization of geometry and material as well as transformation of the objects. The objects' position can also be linked to sound sources of the sound field (see Fig. 3).

In addition to the conventional environment module, the toolkit also offers the *IVES.vr_environment* module for using PC-VR systems. This module offers similar functionality as *IVES.environment*, but extends the rendering to HMDs using the VR library. The implementation is using the VR external with multiple cameras for each eye as documented in the package. The VR external also provides position and head-tracking parameters from the VR system. These are converted to the coordinate convention used in Spat and can be transferred to the listener's position and orientation by connecting the module to the *IVES.soundfield*.

2.2.1 *Spatial Parameter Conventions and Conversion*

Because the Spat library uses a different coordinate system convention than Jitter/OpenGL, all position data and parameters must be converted for the communication and data exchange in-between these libraries. The VR library uses the convention implemented in Jitter/OpenGL. Thus, for 6DoF audio spatialization, the tracking data must be converted to the Spat conventions accordingly.

All this conversion is done by the IVES modules automatically and all GUI elements of the modules use the convention implemented in the Jitter library to ensure consistency throughout the modules. The externals provided by the Spat library support parameters in spherical/navigational (azimuth, elevation, distance) and Cartesian coordinates (x-,y-,z-axis). Jitter also supports Cartesian coordinates, but the used convention is different.

Spat uses a right-handed coordinate system with the positive x-axis to the right, y-axis front to back, and z-axis up and down from the listener. The OpenGL Jitter externals are using the coordinate convention of the OpenGL API, a system with the positive x-axis to the right, y-axis up and down, and z-axis back and front from the listener. The Jitter/OpenGL convention was chosen as the leading coordinate convention. As Spat uses the spherical/navigational

Figure 4. Spatial composition in IVES. The example shows a grid generator (IVES.grid) creating circular positions for four virtual sound sources, and the IVES.move trajectories generator controlling the movement of the first sound source. Both modules are connected to the sound field module managing the data of the virtual sound field.

coordinate system by default, a conversion from spherical to Cartesian coordinates into Jitter/OpenGL convention, and vice versa, is implemented. Because Spat is interpreting Euler angles (yaw/pitch/roll) in relation to their Cartesian coordinate convention, they are also adapted to the convention used by Jitter/OpenGL.

2.3 Control and Interaction

To support the process of composition and interaction with the audiovisual environment, IVES also provides modules to create and transform spatial parameters of virtual objects by integrating further externals from Spat aided for spatial composition (see Fig. 4). The *IVES.move* module provides an implementation of Spat's trajectory external. The module allows the selection and parameterization of trajectory algorithms and apply them to sound sources, 3D objects, or the listener/viewer.
IVES.grid provides an implementation of the *spat5.grid* external, providing algorithms to generate different grids of position coordinates for sound sources as used in the *IVES.soundfield~* module. Spatial position coordinates of the elements described in the sound field can be transformed with the *IVES.transform* module, providing an implementation of the same-named Spat external.
The *IVES.soundsource~* module allows the handling of individual sound sources. It provides a sound player with different parameters for playback of sound files. Alternatively, an input signal can be passed through or a noise test signal can be generated. Furthermore, the position in Cartesian coordinates can be parameterized.

3. DISCUSSION AND CONCLUSION

In this paper, we described the first pre-release version of the IVES toolkit [2] . At the current state, the toolkit already provides all basic modules necessary to create a simple audiovisual virtual environment for screen-based or virtual reality applications with loudspeaker- or headphone-based spatial audio reproduction.
While the basic 3D environment, sound spatialization and VR features are already implemented in IVES, the visual world creation modules are still quite limited, especially compared to game engines. Although the future development aims to further develop and expand visual capabilities, the goal is not to create a full-scale alternative to game engines. Instead, the IVES toolkit will concentrate on single scene environments for artistic application, generative 3D visuals and sounds, as well as genuine audiovisual interaction concepts, rather than complex multiple-scene worlds, sophisticated rendering and simulation algorithms or scripted interactions. The idea is to encourage the development of new artistic concepts focused on sound and music in the field of audio-visual 3D environments and VR, without overwhelming the artists with standard features from game development.
As is the case with most software of this type, continuous expansion and further development is intended. Be-

[2] Available: `https://github.com/AudioGroupCologne/IVES/releases/tag/0.1-smc`

sides the usual bug fixes, improvements and simplifications as well as enhancements with different focuses are planned and/or already in development. To enable the creation of more complex visual environments, additional modules for composition with 3D objects are in development. Also, additional interfaces for further controllers and tracking devices, as well as for software for spatial composition (e.g., Iannix [24]) will be created. Future development will focus on modules for composition with the sound fields in the spatial domain (cf. Sec. 2.1). This approach is less explored compared to the composition of simulated sound fields (positions, motion) and offers the potential for new techniques for sound processing, composition, and the development of spatial musical instruments [25]. The most recent and future versions of IVES are available at: `https://github.com/AudioGroupCologne/IVES`

Acknowledgments

The work has been carried out within the project EarKAR, funded by EFRE (European Funds for Regional Development) under the funding reference code EFRE-0801444.

4. REFERENCES

[1] M. A. Baalman, "Spatial composition techniques and sound spatialisation technologies," *Organised Sound*, vol. 15, no. 3, pp. 209–218, 2010.

[2] "Holoedit," http://dvlpt.gmem.free.fr/web/static.php?page=HoloEdit_main, accessed: 2021-03-03.

[3] C. Bascou, "Adaptive spatialization and scripting capabilities in the spatial trajectory editor Holo-Edit," in *Proceedings of the 7th Sound and Music Computing Conference (SMC)*, 2010, pp. 1–6.

[4] "Zirkonium," https://zkm.de/en/about-the-zkm/organization/hertz-lab/software/zirkonium, accessed: 2021-03-03.

[5] C. Miyama, G. Dipper, and L. Brümmer, "Zirkonium 3.1 - A toolkit for spatial composition and performance," in *Proceedings of the International Computer Music Conference (ICMC)*, 2016, pp. 312–316.

[6] "Spat," https://forum.ircam.fr/projects/detail/spat/, accessed: 2021-03-03.

[7] T. Carpentier, "A new implementation of spat in Max," in *Proceedings of the 15th Sound and Music Computing Conference (SMC)*, 2018, pp. 184–191.

[8] "ICST Ambisonics Toolkit," https://www.zhdk.ch/forschung/icst/software-downloads-5379/downloads-ambisonics-externals-for-maxmsp-5381, accessed: 2021-03-03.

[9] J. C. Schacher, "Seven Years of ICST Ambisonics Tools for MAXMSP - A Brief Report," in *Proceedings of the 2nd International Symposium on Ambisonics and Spherical Acoustics*, 2010, pp. 1–4.

[10] M. Wozniewski, Z. Settel, and J. R. Cooperstock, "AudioScape : A Pure Data library for management of virtual environments and spatial audio," in *Proceedings of the Pd Convention*, 2007, pp. 1–6.

[11] "Cycling'74 Max," https://cycling74.com/, accessed: 2021-03-03.

[12] "PureData," http://msp.ucsd.edu/index.htm, accessed: 2021-03-03.

[13] "SuperCollider," https://supercollider.github.io/, accessed: 2021-03-03.

[14] I. I. Bukvic and J.-S. Kim, "μ Max-Unity3D Interoperability Toolkit," in *Proceedings of the International Computer Music Conference (ICMC)*, 2009, pp. 1–5.

[15] R. Graham and S. Cluett, "The soundfield as sound object: Virtual reality environments as a three-dimensional canvas for music composition," in *Proceedings of the AES International Conference on Audio for Virtual and Augmented Reality*, 2016, pp. 1–7.

[16] G. Santini, "Composing space in the space: An Augmented and Virtual Reality sound spatialization system," in *Proceedings of the 16th Sound and Music Computing Conference (SMC)*, 2019, pp. 229–233.

[17] "VR library for Max," https://github.com/worldmaking/vr, accessed: 2021-03-03.

[18] G. Wakefield and W. Smith, "Cosm : a Toolkit for Composing Immersive Audio-Visual Worlds of Agency and Autonomy," in *Proceedings of the International Computer Music Conference (ICMC)*, 2011, pp. 1–8.

[19] "OpenGL," https://www.opengl.org/, accessed: 2021-03-03.

[20] F. Zotter and M. Frank, *Ambisonics: A Practical 3D Audio Theory for Recording*. Cham: Springer, 2019.

[21] "Talk on Spat4Sat, by Fraction / Éric Raynaud," https://medias.ircam.fr/x80fd02, accessed: 2021-03-03.

[22] "binaural_spat, by Kasper Fangel Skov," https://github.com/kasperskov/binaural_spat-v1.0, accessed: 2021-03-03.

[23] D. Dziwis, T. Lübeck, and C. Pörschmann, "Body-controlled sound field manipulation as a performance practice," in *Proceedings of the 148th AES Convention*, 2020, pp. 1–6.

[24] "Iannix," https://www.iannix.org/en/, accessed: 2021-03-03.

[25] A. Pysiewicz and S. Weinzierl, "Instruments for spatial sound control in real time music performances. a review," in *Musical Century in the 21st Instruments*, T. Bovermann, A. de Campo, H. Egermann, Sarah-Indriyati, Hardjowirogo, and S. Weinzierl, Eds. Berlin: Springer Nature, 2017, pp. 273–296.

VIRTUAL ACOUSMONIUM: A STUDY ON EXPRESSIVENESS OF MUSICAL GESTURES

Stefano CATENA (stefano.catena23@gmail.com) [1] and **Andrea BOLZONI** (andreabolzonimusic@gmail.com) [2]

[1] **Independent Researcher**, Milano, Italy
[2] **Conservatorio G. Verdi**, Milano, Italy

ABSTRACT

For the composer and the interpreter of acousmatic music, the ability to place a sound in space has become an essential element in the creative practice. Together with pitch, timbre, intensity and duration, the position of sounds in space determines its role as a whole. Even more, the movement of sound in space allows them to evoke, with the help of the interpreter, representations of physical environments and gestures. The gesture as a source of motion and the movement that follows, allows you to dive fully in the sound environments depicted. The relevance of this aspect has led us to study and to experiment the behavior of a virtual environment, more precisely a virtual Acousmonium, testing on it various gestures that are part of the tradition repertoire of acousmatic interpretation. The goal is to verify its effectiveness as a compositional tool, as a tool for studying, for the preparation of acousmatic music performances, and as a didactic support for interpretative practice.

1. INTRODUCTION

In the acousmatic compositional practice, the concept of *spatiality* has always been a central point: the representation of plausible or unlikely environments, the movement of sources within a physical or virtual space, the evocation of soundspaces, can bring an acousmatic work to a completely different dimension. In particular, the Acousmonium is one of the most used diffusion systems for acousmatic music: it is the so-called "loudspeaker orchestra", where a large number of speakers are strategically placed in space, each with their own frequency response. The performer of acousmatic music has access to the potential of the Acousmonium through the mixing console: with interpretative gestures, he can decide both the movement and coloring the sound, giving its own personal imprint on the music.

The idea of creating a virtual Acousmonium comes in the first place from the difficulty of using a real Acousmonium: it can be complicated to be able to test, perform and experiment on these system, given the size and complexity. This trouble in accessing the system reflects on artists, performers and composers: a work reproduced by an Acousmonium will be very expanded and enlarged. The composers will therefore have to understand, within the structure of the music, how this enlargement can be compatible with their ideas.

For example: if we have two very fast and short sounds in succession, how will they behave positioned in a space ten times wider, played ten times louder? Without the availability of an Acousmonium it is much more complicated for the artist to be able to relate to the space in which he will play, not to mention the fact of not being able to access the control interface.

The creation of a virtual Acousmonium compensates for this problem by making accessible a system that not always can be so. The use of modern spatialization technologies also make this system flexible and usable in different situations in the studio: Ambisonics, in particular, allows to spatialize sources both in binaural and in a multichannel environment. Developed by the engineer Michael Gerzon in the 70s, the Ambisonics system gives the possibilty to encode a sound field taking into account its directional properties [1]. In the case of a virtual Acousmonium, for example, each speaker is represented in the Ambisonics system through its coordinates in a virtual space, from which the physical characteristics of the acoustic field of that source are obtained. The decoding occurs according to the audio reproduction system, in which the number of components is proportional to the level of complexity used in the coding process (called *order*): this can take place in headphones (binaural decoing) or in a multichannel configuration [2].

Furthermore, the possibility of automating and programming gestures and movements for the machine to perform, is extremely interesting: in this way, many fascinating topics can be delved into, such as didactic applications (ear training and acousmatic composition) or performative applications (assisted spatialization, practice tool at the control interface).

2. BACKGROUND

From a non-musical point of view: "gesture is a movement that you make with your hands, your head or your face to show a particular meaning" [1]. Gesture and sound are two intimately related dimensions. The musical gesture is the movement that produces sound, which relates the

[1] https://www.oxfordlearnersdictionaries.com/definition/english/gesture_1?q=gesture

sound object - the object that "plays" - with the cause of the sound itself, inextricably linking music to physical activity [3]. The physical gesture as an "expression of a thought" is therefore linked to a sonic result, characterized by its tonal qualities. In the spectromorphological terms of Smalley, "A gesture is therefore an energy-motion trajectory which excites the sounding body, creating spectromorphological life" [3]. Consequently, the morphology of a sound will bring with it the gestures that produced it.

However, a sound gesture can also be a movement of sound in space. According to Roads [4] we, as human beings, not only memorize and reason in spatial terms, but perceive space. The same emotional charge that we can evoke from the perception of a sound gesture, can be obtained through the movement of sounds in space. The space as a compositional element was used by various composers, from the use of separate choirs in the 16th century in the Basilica of San Marco, to Mozart, Berlioz and Mahler, Stockhausen and Grisey. But it is undoubtedly in electroacoustic and acousmatic music that space comes to have the same relevance as pitch, rhythm and timbre [4]. From the potentiomètre d'espace created by Poullin in the 1950s, to the sound diffusion in electronic music studios of the same years, passing through the panning techniques of voltage-controlled synthesizers up to the potential offered by computers, the use of audio-oriented programming languages or to the implementation of plug-ins in DAWs, space has always been a compositional and performative element of this musical art [5].

Spatiality and spatial gestures have an enormous relevance in acousmatic music and especially in the Acousmonium, the diffusion system created specifically for this music genre. The term acousmatic is derived from the Greek *akusmatikoi* and describes the sound that is heard without identifying the cause. The adjective refers to the lessons of Pythagoras that the disciples had to listen without being able to see the teacher, hidden by a veil. It was the French composer Pierre Schaeffer who first coined the term acousmatic music in his "Traité des objets musicaux" [6]. By isolating the sound from the visual context, acousmatic music returns to the hearing the total responsibility of a perception that normally relies on other sensitive testimonies [3].

The Acousmonium, instead, was born from the idea of Francois Bayle and the Groupe de Recherches Musicales (GRM) after the experiences of concrete music in the 1950s and 1960s by Pierre Schaeffer: they needed to evolve electronic music, exploring all the problems related to listening and the little interest aroused by simple stereophonic reproduction in the early 70s. Inspired by the orchestra as a standardized and unified concept by Haydn, Bayle wanted both to create a predefined framework in which the acousmatic composers could express their ideas, and to define a reproduction system of great impact in terms of sound and timbre. Almost simultaneously, Bayle's first Acousmonium was born at GRM, premiered on February 12th, 1974 in Paris, while Clozier created the Gmebaphone for the Group de Musique Expèrimentale de Bourges in 1973. The latter separated the various audio channels by filtering the original source, sorting the timbrically modified signal to the various speakers; Bayle's Acousmonium, on the other hand, created its large timbral palette by positioning speakers with radically different frequency response in space. This made it possible to circumvent the problem of the signal's phase deriving from filtering of the source, which can lead to big problems of localization of sound and to sound artifacts. Today, compared to the 1970s, there are much more developed supports such as the computer and the possibility of editing channels directly on the mixer. In addition, there are Acousmonium configurations that provide a huge number of speakers, with different degrees of height compared to the listener, or with an impressive variety of speakers. With the creation of such systems, the importance of the spatial interpreters has grown considerably: they are performing musicians, require a certain degree of virtuosity (depending on the speaker system and the ergonomics of the sound projection instrument) and stylistic knowledge of the repertoire [7]. According to Vande Gorne, there are sixteen gestures (or spatial figures) that are applicable to the interpretation of stereo compositions at the Acousmonium: the spatial interpreter is responsible for binding these gestures together in order to reinforce the writing of the work [7].

In 2016 Barret and Jensenius [8] and Kermit-Canfield [9] present two different versions of virtual Acousmonium. However, the work presented here is specifically modeled after the Sator Acousmonium [2], and a number of gestures have been formalized and experimented on this virtual system in order to test its robustness.

3. THE VIRTUAL ACOUSMONIUM

The virtual Acousmonium was developed with the SuperCollider programming language [10] and relies on spatialization libraries based on Ambisonics [1]. The basic idea for the recreation of an Acousmonium was to emulate the desired number of speakers, place them in a virtual space and "color" the frequency response based on the position and type of speaker.

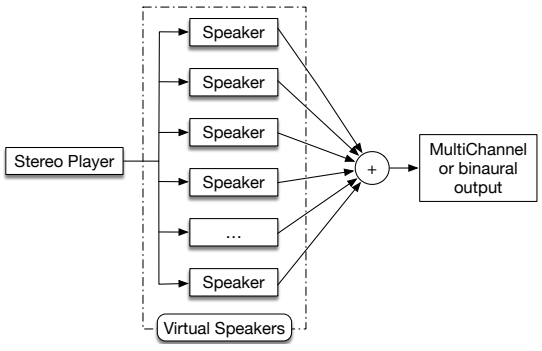


Figure 1. Audio flow diagram from a single stereo source.

Each of these speakers receives a stereo audio signal and,

[2] https://www.centrosanfedele.net/musica/

through a series of equalizations, delays, reverberation and spatialisation, it will emulate the speakers chosen in the virtual space.

The user will have an interface where the sliders of the relevant speakers (or groups of speakers) will be displayed, being able to manually control their amplitude, the buttons to activate the automations and the waveform of the stereo file being read.

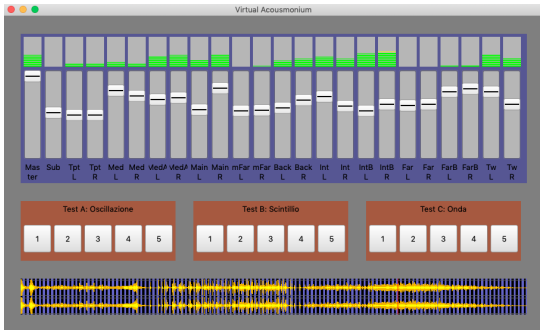


Figure 2. The Virtual Acousmonium prototype GUI.

Ideally it is useful to have MIDI or OSC controllers with a sufficient number of faders, in order to have manual control on the individual sliders and receive visual feedback in real time. In particular, a Korg NanoKontrol [3] and a 16n Faderbank (figure 3) were used for testing.

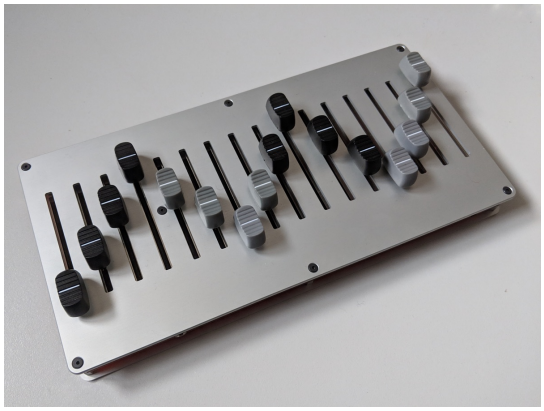


Figure 3. 16n Faderbank.

3.1 The Acousmonium Sator

The virtual Acousmonium implemented here is based on the configuration of the Acousmonium Sator of the Centro San Fedele in Milan [4]. Designed by Eraldo Bocca, the Sator system consists of different types of speakers distributed along three concentric crowns and an effects section which, controlled by a console consisting of a Yamaha LS9 mixer and a Yamaha 03D mixer for a total of 48 channels, allows the diffusion of acousmatic, electroacoustic and mixed music.

Different groups of speakers are already present in the room:

- an internal crown of 9 Nexo speakers + a Nexo subwoofer;
- an external crown formed by four speakers mounted in the corridor of the balcony which produce a reflected and reverberated sound;
- two Db speakers positioned on the balcony for rear reflected sound and eight JBL speakers for cinema surround;
- two JBL main monitor speakers on stage;
- a central JBL speaker on stage;
- a JBL subwoofer speaker on stage.

Speakers built and installed specifically for the completion of the acusmonium are also present:

- two front distance speakers positioned in the backstage;
- a front section of colored loudspeakers positioned on the stage made up of two loudspeakers for the reproduction of dipole emission mediums;
- two diffusers for the reproduction of medium-high dipolar emission;
- two hyperbolic horns for the reproduction of higher frequencies;
- six supertweeters are inserted in the auditorium sky;
- on the balconies four loudspeakers for medium and four supertweeters are installed.

The authors have chosen to emulate this configuration of the Acousmonium given their collaboration with the Centro San Fedele and its availability for testing.

Figure 4. The stage of San Fedele with the Sator's coloured speakers and the Yamaha console.

3.2 Implementation and structure

Given the configuration of the Acousmonium Sator, we opted for a narrower recreation than the large number of speakers actually on site. In total 27 virtual speakers have been created (plus control over the master):

[3] https://www.korg.com/it/products/computergear/nanokontrol2/

[4] https://www.centrosanfedele.net/musica/acusmonium-sator/

- 1 subwoofer;
- 6 coloring speakers: 2 trumpets, 2 medium, 2 medium-high;
- 6 full-range speakers: 2 front, 2 back and 2 front echo effect;
- 4 full-crown speakers: 2 sides and 2 back;
- 4 external crown speakers (echo effect): 2 sides and 2 back;
- 6 supertweeters.

These virtual speakers are placed in space in accordance with the arrangement of the Acousmonium Sator. In particular, to create the echo effect, a great distance from the listener is emulated through delay time, reverberation, and amplitude scaling (figure 5), just like in [11].

```
1  SynthDef(\src , {
2      arg x = 1, y = 0, lpf, hpf, amp, d, eq1, db1, eq2, db2, eq3, db3;
3      var sig, enc, dec, out, rev, del, eq;
4      //x and y positions between -pi/4 and pi/4
5      x = x.linlin(-1, 1, -pi/4, pi/4);
6      y = y.linlin(-1, 1, -pi/4, pi/4);
7
8      sig = In.ar(~srcBus, 2)*amp; //input signal
9
10     //speaker equalization
11     //low and high frequencies boundaries
12     sig = LPF.ar(HPF.ar(sig, hpf), lpf);
13
14     //3 bands parametric EQ with individual gain
15     eq = BPeakEQ.ar(sig, eq1, 0.5, db1);
16     eq = BPeakEQ.ar(eq, eq2, 0.5, db2);
17     eq = BPeakEQ.ar(eq, eq3, 0.5, db3);
18
19     //encoding and binaural decoding with x,y positioning
20     enc = PanAmbi3O.ar(sig, x, y);
21     dec = BinAmbi3O.ar(enc);
22
23     //delay and riverbero related to speaker distance
24     del = DelayN.ar(dec, 0.3, d.linlin(0, 5, 0.001, 0.3));
25     rev = FreeVerb.ar(del, 1, 0.8, 0.8);
26
27     //output stage
28     out = (dec*(1/d)+(rev*(1/d.sqrt)));
29     Out.ar(~revBus, out);
30 }).add;
```

Figure 5. SynthDef of a virtual speaker spatialized with binaural Ambisonics 3D reproduction [12].

To differentiate the virtual speakers, the most important aspect is equalization: in figure 5 (from line 12) the signal is firstly delimited in the high and low frequencies (with LPF and HPF) and then a 3-band equalization (BPeakEQs in line 15-17) with individual gain is applied. In particular, the extreme limits of the signal are important in order to denote the speakers: for example, the tweeters have a range from 4,000Hz to 20,000Hz. This is even more critical in the so-called "colouring" speakers: the trumpets have a range from 3.500Hz to 10.000Hz, the middle ones from 300Hz to 800Hz while the medium-high ones from 600Hz to 3.500Hz. To give the sound more liveliness, the frequencies and gains of the 3 parametric equalizers for each speaker are slightly randomized, creating a phasing effect also between the L and R channels for each family of speakers.

The subwoofer plays a separate role: for convenience, the virtual speaker dedicated to low frequencies is considered omnidirectional, therefore not spatialized. Furthermore, a convolution reverber is applied to the final mix: since the room of San Fedele has a very "dry" and little reverberant acoustics, the impulse response applied to the virtual Acousmonium is one recorded in a medium size concert room with a percentage of "wet" around 20%.

The spatialization of each virtual speaker is made with third-order Ambisonics, both for binaural rendering and for quadraphonic reproduction. These two solutions are particularly suitable in the studio, as Ambisonics is a technology that can quickly switch between the two decoding techniques but that requires the listener positioned in a very well-defined sweet spot [2] (in quadraphony).

One of the main problems of Ambisonics is also the poor directionality. Ambisonics of the first order encodes each sound field as an omnidirectional microphone (zero order microphone) or as a microphone with 3 figure-eight (first order microphone). It is possible to improve the directional sound of the spatialized sound by using higher order microphones: these techniques are called HOA (Higher Order Ambisonics) [2].

4. THE MUSICAL GESTURE IN SPACE

In musical research, movement has often been linked to the concept of gesture. The reason is that many musical activities (performance, conduction, dance etc.) involve body movements that evoke very precise meanings: these movements are called *gestures* [13].

There are many ways in which body movements related to musical contexts can be treated, measured, described and applied. Consequently, there are many ways in which the musical gesture can be meaningful. Musical gestures can also be recognized within a context of spatial interpretation at the Acousmonium. The authors propose two categories of primary gestures:

- Compositional gestures: the musical gestures within the work that can be observed spectromorphologically, that is, by studying how the sound is articulated over time timbrically, dynamically, rhythmically;
- Interpretative gestures: the musical gestures and spatial movements produced by the acousmatic interpreter during the performance.

These two categories are closely correlated: compositional gestures very often correspond to an interpretative gesture aimed at accentuating or highlighting certain intrinsic movements in the work. To these categories can be added other types of secondary gestures: for example physical movements of the performer as an aid to memorization.

It is also important to note that the term *spectromorphology* is not to be considered as an objective and scientific concept, as a scientific analysis of sound is not very interesting, although the latter can help in discovering some details of the sound itself: it is much more interesting, from an artistic point of view, the sound as perceived by the human ear. This is because the listener instinctively responds

to body energy and the spectral qualities of sound: this is what happens also during tonal music performances, as it is not only a matter of pitches, harmonies and rhythms, but also of timbre or spectromorphological qualities [3].

4.1 Gestures in acousmatic music and gestural sonorous objects

Although Pierre Schaeffer is more commonly associated with the concept of *concrete music* and *acousmatic listening*, one of his great successes was the idea of the *sound object*. The sound object is a fragment of a sound, typically of a few seconds or even less. It allows to have a vision of an entire fragment of sound represented with a shape, therefore an object, with different characteristics simultaneously evolving between the initial and final points of the latter (timbre, dynamics, texture etc). These objects are raw fragments of sound, some of which can be chosen and used within musical compositions, and therefore elevated to the status of musical objects.

Schaeffer, in his monumental "Treatise on musical objects", observes how the sound object is an intentional unity, constituted in our consciousness by our mental activity [6]. The sound object can be inspected, explored and progressively differentiated through its own characteristics, which evolve or have different envelopes that can be traced, becoming, according to Godøy [14], the gestural object. Godøy says that there is a continuous mental sound tracking in the musical perception following the onsets, contours, textures, envelopes etc. with hands, fingers, arms or any effector organ (capable of responding to nervous stimuli) whenever we listen to or imagine music. This means that from listening or continuously tracking the sound we can recode musical fragments into multimodal gestures based on biomechanical constraints: in short, we move our body according to the type of sound to which we are subjected. The opposite can also happen, that some gestural images can generate sound images: it is therefore a two-way process.

In the case of acousmatic music, the listener cannot see what the gesture that produces the original sound is: one of the key concepts in the theories of listening to acousmatic music is precisely the fact of putting aside the anecdotal causes or meanings of sound, but focusing only on the intrinsic characteristics of the music. In any case, it is quite clear that Schaeffer, in his studies, made great use of gestural and metaphorical concepts in qualifying the sound objects. Again according to Godøy [14], Schaeffer's use of gestural concepts and metaphors can relate to an idea of embodied cognition, in which virtually every domain of human perception and thought (even the most abstract ones) can be connected to images of movements. The concept of sound-gestural objects can be introduced, an extension of the sound objects in relationship to the gestures that it is transmitted to us during the acousmatic listening. In Schaeffer's works, gestures can be divided into three caterogies of components concerning sound production:

- impulsive or discontinuous type;
- sustained or continuous type;
- iterative type;

Schaeffer also defines other categories of gestures, called compound (where multiple sounds start simultaneously) and composite (multiple objects merged together into one). Furthermore, these gestures relating to the production of sound match the different spectromorphological categories (always defined by Schaeffer) like, for example: changes in mass or harmonic timbre, but also in dynamics, in melodic profile (general changes in pitch) or in mass profile (changes in the intrinsic spectral content), all caused by changes in speed, pressure, direction of the original production gesture. The gestures concerning the modification of the sound are also combined with these morphological categories: modulation gestures such as the application of vibrato or tremolo at different amplitudes and speeds, but also changes in mass, dynamics, profile of pitches etc. previously mentioned. The point is to show that there are gestural components incorporated in Schaeffer's conceptual apparatus and inside his compositional works.

These gestural components can be applied to different sounds interchangeably: a tremolo can be applied to both a violin sound or to the sound of the pouring rain. We can therefore say that they have a certain degree of abstraction: they are transferable from one "domain" to another, both at the physical and at the musical level. In this case, all the characteristics of the sound-gestural object can also be applied to the concept of space and spatiality, in particular to the acousmatic interpretation and to the Acousmonium, as it is strictly related to the spectromorphology of sound and its projection in a diffusion space.

5. EVALUATION OF THE VIRTUAL ACOUSMONIUM

5.1 Tested gestures

To test the virtual Acousmonium, some of the gestures that Annette Vande Gorne [7] identifies were used. Through the automation of the faders, two gestures have been coded in order to test the system:

- **Le fondu enchaîné** (crossfade): slow or imperceptible transition between two pairs or groups of speakers. The gesture must be carefully performed in order not to dig a "sound hole". Between the two groups of speakers, begin to gradually increase the first one, decrease the second one finding a balance point. Musical function: reinforcement of a crossfade already pre-existing in the work. Change depth plane. Trace a path through successive crossfades if, for example, this sound evokes a moving object (ball, car, plane, etc.);
- **La vague** (wave): round trip that crosses, through cross fades or subsequent unmasking, a series of inline speakers, for example from the backstage towards the front stage, all sides, the back. Musical function: moving mass effect and predictable unidirectionality. This movement has the advantage of joining a known agogic archetype.

5.2 Interpretations

The tests were carried out by automating five chosen interpretations of each gesture in the Virtual Acousmonium with binaural spatialization: these automations are generated through different combinations of fade movements that closely approximate the desired interpretation of the gesture.

The tested gestures are *Crossfade* and *Wave*. The interpretations, described through macro areas of the Virtual Acousmonium, are as follows:

Crossfade (crossfading areas reported):

1. back left, near right - back right, near left;
2. far right - far left;
3. sides far - sides near;
4. back left, near back right - back right, near back left;
5. front near - front far.

Wave (paths reported):

1. front far, front near, sides near;
2. back, all sides, front near;
3. far back, back, near back;
4. far, sides near, above;
5. left, front, right;

5.3 Preliminary evaluation

In order to collect data for future developments, we run a preliminary evaluation with five students in electronic music composition, one Bachelor's and four Master's. They have various degrees of experience with acousmatic music, but all can be effectively considered as expert listeners.

Every tester run the evaluation with their own equipment. They were sent three of the five interpretations for each gesture, six audio files in total, and a questionnaire. The audio files were labeled with the name of the gesture and an incremental number (e.g. Crossfade 1). The questionnaire reported the gestures' descriptions, as in Paragraph 5.1, and a list with the five descriptions of the interpretation as in Paragraph 5.2. For each audio file, we asked them to *i)* pair it with one of the interpretations, and *ii)* evaluate their confidence in matching the audio with the description through a Likert scale from 1 to 5, where 1 is the minimum and 5 is the maximum of confidence.

These gestures have been tested on an acousmatic composition of one of the authors. The binaural audio of each of the interpretations is accessible on GitHub [5] .

Every interpretation has been evaluated by three testers. In Table 1 and Table 2 we report the number of correct matches for each interpretation (min = 0, max = 3), and the average confidence for the correctly matched ones (min = 1, max = 5).

The results obtained from the virtual implementation of the Acousmonium can be considered moderately satisfactory: in particular, the movements that include the side speakers (internal, external crown and far/near effect) are very effective and have been recognized consistently by the testers.

[5] https://github.com/StefanoCatena/VirtualAcousmonium/tree/master/Audio

Crossfade	Correct	Confidence AVG.
1	1/3	2
2	2/3	4.5
3	1/3	2
4	1/3	4
5	1/3	4

Table 1. Evaluation of the five interpretations of the gesture *Crossfade*. Correct matches (0 to 3), and average confidence (1 to 5).

Wave	Correct	Confidence AVG.
1	1/3	5
2	0/3	//
3	0/3	//
4	0/3	//
5	3/3	3.7

Table 2. Evaluation of the five interpretations of the gesture *Wave*. Correct matches (0 to 3), and average confidence (1 to 5).

Gestures that include a front/rear movement, instead, have been confusing for the testers, as shown by the results of their questionnaire. The unclear distinction between back and front, however, is typical of binaural technology: this confusion is often a problem of these renders due to ambiguous interaural cues and therefore relying only on monoaural spectral differences [15]. Moreover, given the individual morphological difference in pinnae, each tester would experience the front/rear movements differently: this problem can be minimized by choosing the correct binaural render for the corresponding pinnae's shape. Some testers have pointed out how it is harder to recognize certain movements at first glance, while it takes some time to get used to the binaural rendering. This is especially true for the front/rear movements: while the directionality of the movement has been easier to identify, it was much harder to recognize the difference between front and back. This is also shown by the fact that all testers have misidentified Wave 1 with Wave 4 and, in fact, nobody has correctly recognize Wave 4. This can be attributed to the similarities in the gestures, but also on the fact that both rely on frontal movements and are prone to confusion with rear movements. It is also noticeable how all the testers have recognized correctly Wave 5, where only lateral movements are present.

6. CONCLUSIONS AND FUTURE WORKS

This paper presented the development of a virtual Acousmonium and a first approach to the study and automation of musical gestures in space applied to this system. This tool is not intended in any way to replace the Acousmonium, but to be a tool to help students, teachers and acousmatic composers interested in this area of interpretation.

In addition to interesting compositional applications, the

possibility of automating gestures allows the system to be configured also for educational purposes. In particular, it can be effective for:

- training in the listening of gestures, in the form of ear training;
- the possibility of imitating and practicing gestures, displaying them through its GUI, with the aim of acquiring them for later use in performance.

In this sense, the use of this system by students who wish to learn more about the practice of acousmatic performance may be desirable, especially if they are unable to access the physical Acoumonium. In this case it is extremely important to configure the virtual Acousmonium with its control console as similar as possible to the real life performance situation: the arrangement of the faders and which speakers they control, the desired spatial configuration, the amount of reverberation etc.

From an implementation point of view, the use of Ambisonics to replace real speakers is a significant limitation: it is difficult to simulate the irradiation diagram and the directivity of the individual speaker. Physical speaker cabinets have nonlinear radiation pattern, which are impossible to reproduce in a virtual environment with Ambisonics [9].

In the future other improvements will be introduced: the implementation of a spectrogram and general improvement of the GUI, the implementation of a unified system for the creation, editing and use of virtual speaker configurations, a tool for visualizing the speakers in the space for visual feedback; the possibility of recording a performance, or parts of a performance, so that it can be objectively listened back to; possibly to intelligently automate a whole performance.

In addition, a more thorough testing of the system will be performed: a larger number of testers, different binaural renders and new gestures and movements. It will be of relevant interest to realize these tests with quadraphonic or octophonic reproduction systems, in order to evaluate the Virtual Acousmonium's efficiency in more diverse situations as well.

7. REFERENCES

[1] R. K. Furness, "Ambisonics-an overview," in *Audio Engineering Society Conference: 8th International Conference: The Sound of Audio*, May 1990.

[2] D. Arteaga, "Introduction to ambisonics," Escola Superior Politècnica Universitat Pompeu Fabra, Barcelona, Spain, 2015.

[3] D. Smalley, "Spectromorphology: explaining soundshapes," *Organised Sound*, vol. 2, no. 2, pp. 107–126, 1997.

[4] C. Roads, *Composing electronic music: a new aesthetic*. Oxford University Press, 2015, ch. 8.

[5] P. Manning, *Electronic and computer music*. Oxford University Press, 2013, ch. 23.

[6] P. Schaeffer, *Treatise on Musical Objects*. Le Seuil, December 1966.

[7] A. Vande Gorne, *L'interprètation spatiale. Essai de formalisation mèthodologique*. Universitè de Lille, December 2002.

[8] N. Barrett and A. R. Jensenius, "The 'virtualmonium': an instrument for classical sound diffusion over a virtual loudspeaker orchestra," *The International Conference on New Interfaces for Musical Expression*, pp. 55–60, 2016.

[9] E. Kermit-Canfield, "A virtual acousmonium for transparent spaker system," *Proceedings of the Sound and Music Computing Conference*, pp. 233–237, 2016.

[10] S. Wilson, D. Cottle, and N. Collis, *The SuperCollider Book*. Cambridge, Mass.: The MIT Press, 2011.

[11] J. Chowning, "The simulation of moving sound sources," *Computer Music Journal*, vol. 1, 06 1977.

[12] M. Noisternig, T. Musil, A. Sontacchi, and R. Höldrich, "3d binaural sound reproduction using a virtual ambisonic approach," *IEEE International Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems, 2003. VECIMS '03. 2003*, pp. 174–178, 2003.

[13] A. R. Jensenius and M. M. Wanderley, *Musical gestures: Sound, movement, and meaning*. Routledge, 2010, ch. Musical gestures: concepts and methods in research, pp. 24–47.

[14] R. I. Godøy, "Gestural-sonorous objects: embodies extesions of schaeffer's conceptual apparatus," *Organised Sound*, vol. 11, no. 2, pp. 149–157, 2006.

[15] M. Frank and F. Zotter, "Simple reduction of front-back confusion in static binaural rendering," in *Fortschritte der Akustik, DAGA*. DAGA, 2018.

SQUIDBACK: A DECENTRALIZED GENERATIVE EXPERIENCE, BASED ON AUDIO FEEDBACK FROM A WEB APPLICATION DISTRIBUTED TO THE AUDIENCE

Gianluca ELIA[1] and **Dan OVERHOLT**[1]

[1] *Sound and Music Computing*, **Aalborg University**, Copenhagen, Denmark

ABSTRACT

Squidback is a participatory and contemplative experience, a collective generative soundscape without a central preferred point of view, whose sound sources are the audience's smartphones or computers working as audio feedback generators. The work aims at creating a ritual space to explore fields of play between being performer and audience, situating control, affect and listening in between human/machine and machine/environment ecosystemic interactions.

Squidback is implemented as a browser-based app, hosted on the Internet as a perennial web installation [1]. The code is open-source and available online [2]. Evaluation has been done by examining activities in which the system has been used, and comments from artists who included it in their works.

1. GENERAL CONCEPT

Squidback is a technological system and a concept for a participatory performative installation. Its generative process is based on audio feedback (also known as Larsen Effect [3]). Thus, it is naturally responsive to everything surrounding participants' devices, from their acoustic environments to objects and people nearby. It features a custom adaptive filter that adjusts itself autonomously, thereby avoiding users' direct interaction with parameters. Instead, it promotes a contemplative attitude, inviting them to find other ways to affect the process, for example by moving in the room, by creating shapes with their hands around the device, or by approaching other participants and their devices.

No centralized control strategy is implemented: the participants/devices become an ensemble of independent instances of the same process, each giving different results and thus composing a collective, generative, spatialized soundscape. Furthermore, sound spatialization and feedback are in mutual interplay through the decentralization of the system: participants' movements affect sound generation, which in turn affects the spatialized soundscape even if the participants are still; by moving, participants change what they hear (which region of the collective soundscape) and the sound they produce, realizing a further layer of mutual influence between collective and individual dimensions.

Squidback has been originally used as a native smartphone application, for performances where participants shared the same room, and later was made available as a web application, featuring remote online participation. It is thus currently possible to combine these settings by having groups of people, each sharing the same space, performing together online.

2. BACKGROUND

This work was first developed as part of the first author's practice-based research project *Becoming Program, Becoming Performance* at the Rytmisk Musikkonservatorium, Copenhagen (Aug 2017-Jun 2019), which focused on designing and performing with different systems: computer programs, machines, ensembles of musicians and directions for improvisation; in composition, improvisation and production settings. It binds together the main topics informing the first author's general research frame: generative music, decentralized systems, sound in space, and relations between acting and listening during performances.

Squidback was at a later stage ported to a Web Audio application for two reasons: to exit from smartphone-native apps' commercial distribution circuits, and to unify the codebase, ceasing to have two different versions for Android and iOS.

Finally, a real-time audio sharing feature was implemented (using WebRTC [4]), to allow participants to perform together without being in the same room. This opened new possibilities in terms of Squidback's significance as an experienced, reflecting more of the ongoing cultural transformations connected to digital life and communications.

3. RELATED WORKS

The presented work relates closely to three categories of past works: feedback-based resonant assemblages [5] [6], smartphone-based participatory techniques [7] [8] and ecosystemic works [9].

3.1 Feedback generation

As a system and performative installation concept, Squidback fits into the description of *Hybrid Resonant Assem-*

blages coined by Bowers and Haas [5], which features: involvement of different materials and media (sound, lights and objects/textures in the room); *immanent* sound generation (feedback); *transient* performative gestures (i.e. the room-system's construction, deconstruction and exploration) inviting to a gathering and to rethink wider notions of touch and instrumentality.

Squidback's sound process is based on feedback suppression systems (a survey is provided by Waterschoot and Moonen [10]), but instead of completely eliminating feedback frequencies upon detection, exploiting them as musical material. With the piece *Pea Soup* [6] Nicolas Collins was doing this already in 1974, using at first dedicated hardware, and then moving to software emulations. The piece is closely related to Squidback, because it was also produced in both concert and installation format. The difference with Squidback is the array of audience-owned devices, creating a spatialized sound system, bringing the process closer to participants and breaking down the boundary between performers, audience and installation as a completely autonomous and self-standing entity.

3.2 Smartphone-based participation

Among works for smartphone, we can distinguish between implementations which envision devices as instruments for performers to play (like much of the works from *Stanford Mobile Phone Orchestra* [8]) and others that are meant to be run by the audience, almost always including some form of centralized orchestration, or networked operations (like Tate Carson's *A More Perfect Union* or Andrey Bundin's *Concert For Smartphones*). A survey of smartphone-based audience participation strategies is provided by Oh and Wang [7], focusing on the relationship between audience and a "master performer", with audience-audience communication as an emergent property.

Compared to these works, Squidback's approach stands for a decentralized aesthetics, whose unifying force and compositional effort is the development of a singular system that will be run by independent instances, these affecting each other only by sending and receiving sounds through the rooms and/or the Internet. Distributed music as a performance practice has been reviewed by Taylor [11].

3.3 Ecosystemic organization

In his inspiring article "Sound is the Interface", Agostino Di Scipio [9] defined an ecosystemic approach to interaction which differs from the most widely implemented paradigm, turning compositional attention from *interactive composing* to *composing interactions*, and from a question of exerting the proper control over a separate sound generator to the interrelationship between system and environment. The topic has been further elaborated by Pirró [12], first by considering such mutual influences as a central cognitive mechanism, in relation to enaction, and on the other hand through the mathematical language of dynamical systems.

Fitting in Di Scipio's definition, Squidback is an ecosystemic work as much as it is "a dynamical system exhibiting an adaptive behaviour to the surrounding external conditions, and capable to interfere with the external conditions themselves", where man/machine interactions are situated in a system of machine/environment ones. In avoiding centralized control and control interfaces, Squidback reduces the predominance of humans as control agents, allowing the participants more explorative and contemplative roles. However, human activity is still a central component in this work's performative concept, as it is left to the participants to decide both their degree and mode of activity while listening to and exploring the performative space.

4. DESIGN

The main goal of the current design is to generate a variety of frequencies from feedback, avoiding a single frequency becoming dominant for too long, while maintaining ecosystemic interactions within the space where the system is situated. This achieves a balance between its autonomy and users' physical agency. The feedback process is controlled through a bank of peaking filters, wherein individual band gains are automatically adjusted according to the balance of the incoming sound's spectral magnitudes over time. Not being a purely technical problem, it is approached empirically, designing and tuning the process and its reactivity, to follow aesthetic intuition in dialogue with technical insights and theories.

On a high level, Squidback as a single process is depicted in the graph in Fig.1. Sound read from an input device is analyzed for frequency magnitudes, which are stored into a history buffer recording magnitude values from the last M analyzed frames. This historical data is used together with statistics about the most recently analyzed spectrum, to compute individual band gains of a bank of filters processing the input. The last stage of the process is an automatic overall gain adjustment, after which the processed sound is fed to the output mixer, where it's sent to remote peers, and is mixed with their incoming sounds. The choice of not feeding sound from remote peers to the analysis and filtering process is meant to keep the algorithm focused on local input (especially the auto-gain), otherwise sounds from remote peers would decrease its reactivity to the local user's agency.

Naturally, putting a filter bank in between a feedback chain is adding feedback to feedback, thus affecting the generative process. In other words, the system becomes an important part of the room, and it was not a focus of this work to tell apart the instrument (Squidback, and the device itself) from the "measured phenomenon" (the room and its resonances).

5. TECHNICAL IMPLEMENTATION

The application logic was developed in Javascript, using only the Web Audio API [13]. At the time of writing, Web Audio's most modern features, namely Web Assembly and AudioWorklets, were supported only by the latest versions of some major browsers. Therefore, a choice was made not to rely on them.

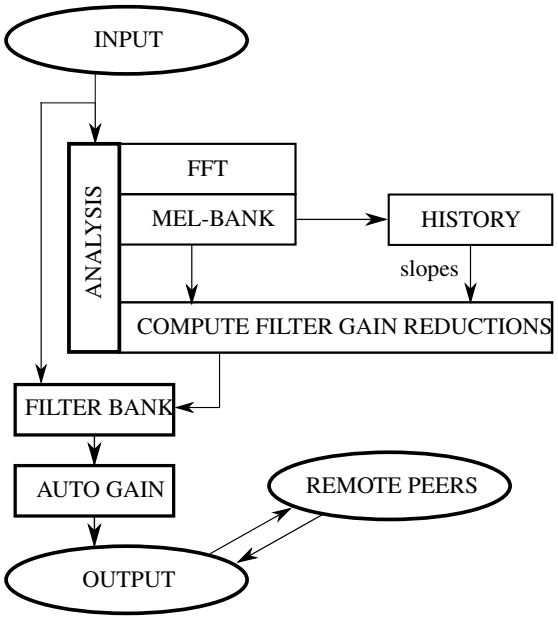


Figure 1. Squidback process diagram

DSP parameters and control strategies were chosen and tuned empirically through several tests on different devices, in accordance with the first author's aesthetics. As Squidback doesn't offer any parametric control interface, all values are hardcoded, and the following subsections refer to the setup chosen at the time of writing, which may well be subject to changes.

5.1 Analysis

5.1.1 FFT

Spectral analysis is performed by calculating the input's FFT, using Web Audio's AnalyserNode. Window size was set at 2048 frames, to provide enough perceptual resolution at low frequencies. Web Audio does not overlap frames, but linearly interpolates magnitudes over time by a factor of 0.8 (default) [14].

5.1.2 Mel-Frequency Filter Bank

In order to more closely match the system's reactivity to a perceptual dimension of pitch, FFT magnitudes are then passed through a bank of triangular filters, to compose their values into a number *nFilters* of mel-frequency bands, within a frequency range between *minFreq* (twice the lowest fft bin's center frequency) and *maxFreq* (10kHz). Although *nFilters* could be variable, a fixed number of 30 filters was chosen to provide some variety and still not be too computationally heavy, especially on a mobile device's CPU. The effect of mel-frequency mapping on the process' reactivity is a mitigation of the over-representation of high pitches, due to the linearity of FFT bin center frequencies distribution. Mel-frequency mapping was chosen over Constant-Q Transform (CQT) because it was found to be easier to implement in Javascript without relying on WebAssembly, like Javascript CQT implementations [1] commonly do to apply Brown and Puckette's spectral kernels method [15].

5.1.3 Reduction calculation

On the most recent mel-frequency spectral frame, minimum, maximum and average magnitudes are calculated. Then, for each bin *k*, a magnitude difference is calculated:

$$\Delta_k[n] = average[n] - magnitude_k[n] \tag{1}$$

Where *n* is the index of the most recent analyzed frame. Then, the system takes into account spectra from past frames, to calculate different corrections whether each band has been increasing, decreasing or stayed within a small range of magnitude values. Historical data is recorded as a weighted average of successive magnitude variations for each band. Magnitude differences across the last M frames (slopes) are considered to be zero (and the band to be constant) if their absolute value is less than a chosen threshold. Then each band gets a score depending on the sign of its slope: on each frame, if the band wasn't constant, its score increases or decreases by one corresponding to whether its magnitude was rising or falling. For constant bands, the cumulative score increases by the opposite of its own sign, bringing it one step closer to 0. Deltas are then adjusted according to the score, in order to stabilize the intensities of each band.

$$\Delta_k[n] = \begin{Bmatrix} 0.5\Delta_k[n], & \text{for } score_k[n] > 0 \\ -0.5\Delta_k[n], & \text{for } score_k[n] < 0 \end{Bmatrix} \tag{2}$$

Before the last calculation phase, coefficients are prevented from being positive, so as to disallow positive gains, and linearly smoothed by a factor λ:

$$\begin{gathered} \Delta_k[n] = min(\Delta_k[n], 0) \\ reductions_k[n] = \lambda(\Delta_k[n-1]) + (1-\lambda)(\Delta_k[n]) \end{gathered} \tag{3}$$

It was found that the system tends to resonate at a small number of dominant frequencies, if nothing changes in its environment. So, to help the system to change shape over the time of a performance, prioritizing the emergence of a variety of new tones, it is beneficial to gradually penalize frequencies that have already been reduced, even further. Therefore, a fraction *m* of the computed reduction is stored in memory as a persistent correction, independently of live spectral data, so that each band that has been reduced accumulates a trace of this activity.

$$\begin{gathered} reductions_k[n] = reductions_k[n] + mem_k[n] \\ mem_k[n] = m(reductions_k[n-1]) + mem_k[n-1] \end{gathered} \tag{4}$$

Finally, the minimum reduction across all frequencies is calculated, to remove any constant gain factor applied to all frequencies, thus letting the filter work more on frequency

[1] https://github.com/mfcc64/showcqt-js

balance than on absolute values. This is achieved by calculating the total frequency response of the filter bank, and subtracting its maximum value from each gain.

5.2 Filter

A bank of peaking biquad filters is created by using Web Audio's BiquadFilterNode, corresponding to the frequency scale used by the mel-frequency part of the analysis process. The first, lowest frequency filter, is set to be a low-shelf type, and the last, highest frequency filter is set to high-shelf. For aesthetic reasons, only reductions are allowed, as it was empirically found that gains applied to individual bands would result in smoothing out too much of the desired Larsen effect's roughness.

5.3 Auto Gain

At the end of the chain, gain applied to the signal is controlled by monitoring the signal's amplitude, so that the magnitude of the loudest bin in the analyzed spectrum, smoothed across subsequent frames, approximates a set threshold. At the time of writing, the smoothing factor was set to 0 (no smoothing), with a threshold at -20 dB and a maximum possible applicable gain of 20 dB.

5.4 Remote Participation

Remote connections are achieved through WebRTC [4], where each client is a peer in a mesh network, and a server is used only to facilitate discovery among clients. Connections among clients are then peer-to-peer.

Figure 2. Visualization

5.5 Visualization

Squidback has a visual output: it displays spectral magnitudes as analyzed by the mel filterbank, overlayed with corresponding filter gain reductions. The current implementation draws a white rectangle originating from the bottom of the screen representing energy in each spectral band, and a black one from the top representing reductions for each filter. Filter graphs are obtained as frequency responses for each band, and their total sum, through Web Audio's BiquadFilterNode.getFrequencyResponse(). The background is colored depending on pitch class [2] and the octave of the loudest bin in the mel-spectrum. While other options (such as more or less smoothed curves) were also implemented, the choice of using rectangles is both for efficiency and aesthetics reasons. To this regard, the visualization provided is not conceived to attract users' attention too much, but to provide some intuitive insights into how the system works, as a complementary experience during performances.

[2] https://en.wikipedia.org/wiki/Pitch_class

6. PERFORMANCE

Squidback can be played as a participative performance, which participants join at the same time, or as an installation, which participants can enter, join, and leave as they wish. Both forms can be played by participants sharing the same physical space, or online on the Internet, or in any combination of the two.

6.1 Physical

A typical performance starts with participants gathering at the entrance of the chosen performative space, which can consist of a single or multiple rooms, preferably indoors to exploit the reflective properties of closed walls. To begin, the first author typically explains how the system works and how to get it and run it. It is important to briefly inform participants of what feedback is, in order for them to form an intuition of how it works, and what they might do to affect it by moving the device and acting on their physical environment. If there aren't strict requirements about the performance's duration, the participant are told that they can stop the process and/or leave the performance whenever they want; otherwise participants will be informed that when it's time to stop, they will receive a signal from either a person or a change of lights in the room. A typical performance duration is between 20 and 60 minutes. After the spoken introduction, participants are invited to explore the performative space and to choose a location where to start the application. Spaces can optionally be prepared with speakers to which participants can connect their devices if they want, to enjoy a wider and/or louder output spectrum, or with objects/musical instruments to interact with. Albeit not strictly required, such setups can offer additional performative approaches to participants, and associate Squidback more tightly with the specific place where it's being performed, by including whatever objects are available there.

6.2 Online

As mentioned in section 5.4, peer-to-peer audio sharing over the Internet can be used to run networked Squidback performances. It has been found beneficial in this situation, to gather participants on a third-party web conferencing platform immediately before the actual performance, to give them an introductory explanation and eventual assistance as has been done in "physical" performances. It is not strictly necessary though, as instructions can be embedded on Squidback's website. Such performances can be described as participatory live-streaming concerts, where participants are at the same time attending to a shared process over which they can exert limited control, and more or less intentionally and recognizably influencing it. Sound

contributions are also richer and more differentiated when participants join from their private space, allowing themselves to use more of their voices, sounds from other media (e.g. televisions, radios, music players) and even musical instruments.

6.3 Installation

As a standalone physical installation, Squidback is just an empty, dark room, with written indications (distributed as program notes, or present as the installation's description outside the room) functioning as the initial verbal explanation does for a performance. People can come and go, in any number, as they wish, but are invited to start the application before entering the room, to further characterize the installation as a performative occasion.

Squidback is also available on the Internet [1], where it is present as a permanent web installation. Differently from online scheduled performances, users can connect and disconnect at any time, and choose whether or not to share their sounds with other potentially connected users.

Figure 3. First public physical performance

7. EVALUATION, RECEPTION AND OBSERVATIONS

The process of producing artistic work is a dialogue between technological and artistic domains, for which the outcome is not measurable analytically in terms of efficiency/efficacy in solving a technical problem. Outcomes of this work are identified in activities in which the system has been used. Our preferred means of evaluation is to examine the work as an artistic process, observing performance experiences, and the receptions and comments by other artists who incorporated Squidback into their own artworks.

7.1 As a performance

Squidback has been performed several times in its prior implementation as a smartphone-native application. A short video edit from early testing stage [16], and documentation of the first public performance [17] are available online.

During these performances participants exhibited different ways to relate to devices, each other and the space through different degrees of activity, energy, mobility, sociality, collaboration and individuality. Typically, participants begin by actively engaging with their devices, then with other people, the space, its surfaces and objects. After around thirty minutes into a performance, it becomes more contemplative as people often lay down to listen to their device and the environment, intermittently changing position. 45 minutes tends to be an optimal duration, allowing for enough time to explore curiosity, excitement, boredom, relaxation and contemplation.

Each participant's own device is most often kept close to its owner, acting as a 'soloist' voice, thereby being most perceptible against the environment's background. It should be noted that leaving their devices alone somewhere in the room in order to have them 'join the choir' however, is not something most participants have been willing to do spontaneously.

7.2 As an instrument

The first author has been playing with Squidback since its first prototyping stages, performing improvised music solo and ensemble concerts as part of mixed electronic and electro-acoustic setups. In such settings, the system acts as an instrument with an autonomous generative quality, providing drones, harmonies, loud piercing frequencies, and/or a resonant reverberation effects. Its consistency and reliability as a standalone module is a strong standpoint for it to enter relationships with other instruments (e.g. no-input mixers). The lack of parametric control interfaces invites performative actions to undertake a dialogic quality, requiring receptivity for the system's own properties and developments. It also invites exploration of a control space that extends to the physical space where the performance is situated, affected by acting physically on the devices and nearby objects. As both a native and web-based application, Squidback is portable and has minimal technical requirements: a smartphone or a computer is sufficient to play it. Such a simple setup has facilitated its usage not only for staged performances, but also for more spontaneous and less planned performative actions.

7.3 As part of other artists' work

In its first incarnation as a smartphone-native application, Squidback was integrated into works by different artists. The comments below explain how Squidback fit their work and research.

7.3.1 Francesco Toninelli

Francesco Toninelli is a Copenhagen based composer, percussionist and improviser. He writes:

> My experience with Squidback starts right after its release in spring 2019, when I first tried to extend its use with bluetooth speakers and percussion instruments. After some testing I ended up with a system where a contact speaker was connected to the smartphone

> via Bluetooth and laid on the top of a kick drum (placed vertically on a support). This way the feedback generated by the app was affected not only by the room but also by the inner acoustics of the drum (as a combination of drum skin and wooden resonant body): needless to say, it was a very suggestive instrument capable of great complexity and diverse applications, such as installations or live concerts, also because of how easy it could be to access one or more smartphones and carry them around.
>
> The experiments on the instrument thus involved both tuning of the drum and use of different rooms or open spaces.
>
> As the time passed I started to be more and more focused on the harmonic material generated by the instrument, finding patterns of behavior. This brought me to create an installation then presented in Tempo Reale Festival 2020 (Florence, Italy) where three Squidback-kick drums were placed in an open space, as distant as possible from each other, to find a central point where one could hear all of them and create a small walk exploring the interaction of chords and its mutations.

A recording of Francesco Toninelli's work for Tempo Reale is available online [18].

7.3.2 Federico Corsini

Federico Corsini is a musician and dancer, based in Copenhagen. He writes:

> Limitation is invitation.
>
> When there is no control interface, that is the control interface. It comes from punk and improvisation: you play with what is there, you make a performance out of what is available. Feedback has an intrinsic property of uncontrollability, and even if not having knobs is a different thing, you accept both as epistemological truths. And it has another property: sound changes waves morphology and tone depending on dimensions and distances between speaker, microphones and the room, and also on the objects present in the room. My body in between speaker and microphone affects qualities of both my body and sounds.
>
> Squidback allows to focus on the relationships to objects and (their) movements in space. More than focusing on my body movement, I focus on objects' movements within the performative space. Adding a bluetooth speaker gives you now two objects you can move around (the speaker and the phone). I tried to interact with those movement and my body in space, at the different physical levels (floor, standing, objects above my head), accepting and reacting to information created by the feedback process.
>
> I'm fascinated by creating dynamics I've never explored before, or experiences I've never tried or rehearsed, to be explored only during a performance, and not to be studied academically.

A short video from Federico Corsini's work with Squidback is available online [19].

7.3.3 Dasha Lavrennikov

Dasha Lavrennikov is a choreographer who used Squidback in her work during an artistic residency in Copenhagen. The work was an outdoors guided walk which featured a solo performance with Squidback while exploring the surface and depths of a rocky landscape.

> Using the app really opened a place of inquisition - questioning around the notion of feedback, in sound and movement. It was a super rich finding in my research around space - sound / space - movement / sound - movement, this triangle of information in the process of improvisation.
>
> In particular, it felt like the most relevant and possible space of sound-movement manifestation at this point in the walk that I shared... as the possibility to question physical space and architectural space, the infinite and its limits, and how through sound and the body we experience these limits, contours, borders, and what that generates in terms of the concrete and the phantasmagorical.

The work in question was shared with a limited number of people as a private performance, of which there's no publicly available documentation.

8. CONCLUSION

The absence of control interfaces calls the controller paradigm into question, inviting for a more fluid relationship between individuals, the adaptive technological process and the environment. Attention flows through exploration and contemplation, curiosity and experimentation, affect and inspiration, activity and passivity. Each participant can choose a different mix between being more of a performer or an audience at any time, blending these two roles in lack of a clearly defined separation, enabling a diversity of approaches to unfold. The decentralized system also contributes to these dynamic relations by making each participant a creative agent on the collective soundscape, wherein their acoustic situation and movements affect sound contributions and perceptions at the same time. Participants create a multi-faceted soundscape, thereby listening to a particular selection and mix within the ecosystem developed between individuality and interdependence across actions and perceptions.

Squidback also works as an inspiring tool for artistic production, primarily in the fields of installation sound-art and contemporary dance. Its limited scope, generative possibilities and reactivity to space and movement, together with a relative ease of adoption and integration, have been strong points reported by artists who have used it in their works.

8.1 Future directions

Future directions will on one hand attempt to make Squidback a more accessible development platform for experimentation with feedback, by exposing a public API to the browser window so that every part of the process can be controlled via Javascript code from a browser's console, including the ability to plot graphs of historical and computed data. Also, the WebRTC implementation is still very simple, and further research could be dedicated to improving its stability and elaborate on possible processing strategies for sounds from remote peers.

The latest version of Squidback is currently a web application, and is thus able to bypass the two main commercial smartphone App Stores. However, the field of web applications is still heavily influenced by browser implementations, where the major players are the same operating commercial App Stores. As much as the authors are looking forward for more advanced features to be widely supported (primarily AudioWorklet support), Squidback still situates itself in a delicate field, where it is intended to exploit devices that people already have, but at the same time it has a strong dependency on their software updates. It could be interesting in the future to explore the construction of embedded devices.

Acknowledgments

The first author wants to acknowledge help and support received from the research and experimental music community in Copenhagen and at the Rytmisk Musikkonservatorium. Several people helped making this project possible, assisting with testing and enthusiastic support. Special thanks to Morten Carlsen and Torben Snekkestad for inspiring conversations since the beginning of the project and warm support throughout. And special thanks to artists who integrated Squidback in their creative works, in particular musician/dancer Federico Corsini, composer Francesco Toninelli and choreographer Dasha Lavrennikov.

9. REFERENCES

[1] G. Elia, "squidback," https://squidback.xyz.

[2] ——, "Squidback: code repository," https://github.com/elgiano/squidback-webaudio.

[3] A. Larsen, "Ein akustischer wechselstromerzeuger mit regulierbarer periodenzahl für schwache ströme," *Elektrotech. Z., ETZ*, vol. 32, pp. 284–285, 1911.

[4] W3C, "Webrtc spec," https://www.w3.org/TR/webrtc/.

[5] J. Bowers and A. Haas, "Hybrid resonant assemblages: Rethinking instruments, touch and performance in new interfaces for musical expression," in *Proceedings of the International Conference on New Interfaces for Musical Expression*. Culture Lab, New Castle University and Berlin University of the Arts, 2004.

[6] N. Collins, *'Pea Soup' - a history*. Nicolas Collins, 2011.

[7] J. Oh and G. Wang, "Audience-participation techniques based on social mobile computing," in *Proceedings of the International Computer Music Conference 2011*. Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, August 2011.

[8] G. Wang, G. Essl, and H. Penttinen, *DO MOBILE PHONES DREAM OF ELECTRIC ORCHESTRAS?* Stanford, California, USA: Stanford University, 2008.

[9] A. D. Scipio, "Sound is the interface: from interactive to ecosystemic signal processing," *Organized Sound*, December 2003.

[10] T. Waterschoot and M. Moonen, "Fifty years of acoustic feedback control: State of the art and future challenges," *Proceedings of the IEEE*, vol. 99, pp. 288–327, 2011.

[11] B. Taylor, "A history of the audience as a speaker array," in *NIME*, 2017.

[12] D. Pirró, "Composing interactions," Ph.D. dissertation, Institute of Electronic Music and Acoustics, University of Music and Performing Arts Graz, Austria, 2017.

[13] W3C, "Web audio api," https://www.w3.org/TR/webaudio/.

[14] ——, "Web audio api: Fft windowing and smoothing over time," https://www.w3.org/TR/webaudio/#fft-windowing-and-smoothing-over-time.

[15] J. C. Brown and M. S. Puckette, "An efficient algorithm for the calculation of a constant q transform," *The Journal of the Acoustical Society of America*, vol. 92, no. 5, pp. 2698–2701, 1992.

[16] J. Exner, "Squidback early teaser," https://vimeo.com/312155346.

[17] Hipermania, "Squidback first public performance," https://vimeo.com/492522089.

[18] F. T. (recorded by Renato Grieco), "Canti aperti unlocked sounds," https://gianlucaelia.eu/audio/toninelli-temporeale.wav.

[19] L. Shimitzu, "Federico corsini // squidback," https://www.youtube.com/watch?v=2k-OyGCbXJw.

AUDIO PARAMETER MAPPING MADE EXPLICIT USING WEBAUDIOXML

Hans LINDETORP (hans.lindetorp@kmh.se) [1,2] and **Kjetil FALKENBERG** (kjetil@kth.se) [2]

[1] *Department of Music Production*, **KMH Royal College of Music**, Stockholm, Sweden
[2] *School of Electrical Engineering and Computer Science*, **KTH Royal Institute of Technology**, Stockholm, Sweden

ABSTRACT

Sonification using audio parameter mapping involves both aesthetic and technical challenges and requires interdisciplinary skills on a high level to produce a successful result. With the aim to lower the barrier for students to enter the field of sonification, we developed and presented WebAudioXML at SMC2020. Since then, more than 40 student projects has successfully proven that the technology is highly beneficial for non-programmers to learn how to create interactive web audio applications. With this study, we present new feature for WebAudioXML that also makes advanced audio parameter mapping, data interpolation and value conversion more accessible and easy to assess. Three student projects act as base for the syntax definition and by using an annotated portfolio and video recorded interviews with experts from the sound and music computing community, we present important insights from the project. The participants contributed with critical feedback and questions that helped us to better understand the strengths and weaknesses with the proposed syntax. We conclude that the technology is robust and useful and present new ideas that emerged from this study.

1. INTRODUCTION

Audio parameter mapping is a commonly used approach in a wide range of applications spanning from data sonification projects to physical, digital instruments with knobs and sliders. Common for these applications is that variables, e.g., statistical data or the value of a knob on a synthesizer, are used to shape or control the playback of a sound.

Sonification using audio parameter mapping involves both aesthetic and technical challenges and requires interdisciplinary skills on a high level to produce a successful result. The process includes data preparation, sound synthesis, mapping parameters and finally listening and tuning the settings [1] to produce a meaningful result. It also requires an understanding of auditory perception [2], sound design, and musical composition. Studies show that the relationship between the original data and the auditory domain is far from being a simple linear link [3] and even if there have been attempts at trying to formalize ways of describing those relationships [4] there is still a great potential for further research. Various programming languages have been developed aiming at meeting those needs, including CSound [5] Max/MSP [6], Pure Data [7], SuperCollider [8] and ChucK [9]. There are also tools available for creating sonifications like xSonify [10], Toolkit for Sonification [11], SonifYer [12] and SoniPy [13] but most available tools arguably requires both programming and audio synthesis skills.

At The Royal College of Music (KMH) and the KTH Royal Institute of Technology in Stockholm we teach sonification, sound design and sound synthesis to both music producers and engineers, which has led to the development of the new coding environments WebAudioXML [14] and iMusic [15], as well as an online sonification toolkit [16]. Our research projects are often tightly connected with the pedagogical activities of ongoing courses [17] and point out that there is a great potential for audio tools that are easy to use in order for the students to express their creativity rather than stumbling on technical challenges [18].

Many of our courses are aimed at building bridges between art and technology as we want knowledge and practices to be shared between the disciplines. While most of the students at KMH have no prior knowledge of programming, we have been challenged to find ways for them to get into coding as easy as possible. Most of our students have a basic understanding of HTML and as the web has become an increasingly important platform for sharing not only static content but also interactive online audio applications, we have decided to explore Web Audio API [19] as the platform for our experiments. We particularly appreciate that web technology is open source and that the applications we build are cross platform, online accessible and require no installations for the end user.

During our sonification classes, we have also discovered the potential for a standardized way of describing the various parameter mappings used in an application to encourage shared knowledge, flexible program architectures, open plugin structures with common libraries of sounding objects, mappings and data manipulations. We aim for solutions that make configurations and mappings explicit to encourage readability and easy assessment to avoid the "black box"-phenomenon that often can be said about sonification applications.

This study contributes to the community of sound and music computing with a proposal for a descriptive XML syntax for parameter mappings in audio applications in general and for web audio applications in particular. We have developed a working example and aim at a better un-

derstanding of the strengths and weaknesses of such a syntax.

1.1 Background

Since the presentation of WebAudioXML [14], we have supervised more than 40 student projects using the technology with promising results. WebAudioXML is a framework that uses XML to abstract an audio configuration for Web Audio API. The declarative syntax has proven to make Web Audio application development accessible for creators and opens up possibilities for students with little or no prior programming experience to turn interactive, audio application ideas into reality. We have also developed WebAudioXML Sonification Toolkit [16] – an online tool for exploring mappings between statistical data and audio parameters – that has been tested by students with no prior experience in either sonification, programming or audio synthesis. The evaluation points towards the need for the creators to access an audio configuration on a meta-level rather than having to understand and control all low-level parameters in a complex audio object.

The initial focus for WebAudioXML was to offer an XML-syntax that described audio connections and configurations. The first version supported mapping external variables to audio parameter but the solution was limited in many aspects. The mappings were defined individually for all audio parameters, only one external variable could control an audio parameter and the mapping was restricted to one pair of in-values and out-values with only one setting for interpolations. While the old version has proven to inspire creativity, it has also indicated a great potential for new features like the ones presented in this study.

1.2 Design Process

This study builds upon earlier experiences from the development and evaluation of WebAudioXML. It covers a further development of parameter mapping in the syntax and introduces a new variable object that offers a standardized way of specifying parameter mappings including scaling, quantization and conversion of values. We use three cases springing from student projects as a driving factor for the design and document the process of developing the syntax. The code is finally tested and published with online examples [1] and discussed with three technically experienced music artists.

1.3 Design Goals

WebAudioXML offers a simple syntax with a hierarchical structure that aims at shifting focus from the technological to the artistic in the making of interactive audio applications. The intention with the current study is to stay true to the same design goals while introducing more complex and flexible parameter mapping solutions. It shares the logical approach to parameter mappings with graphical environments like Pure Data, it is readable like a text based language as SuperCollider, and it uses a spreadsheet-like approach for data bindings to make mappings more explicit. There are different approaches to the use of XML: One is to use elements for encapsulating all data and another is to use elements for the hierarchical structure but store the data using attributes. In this study we want to stay true to the path set out in WebAudioXML where the audio nodes are represented by elements and their parameter values are specified using attributes. This arguably makes the code more compact and easy to write even if attributes are less flexible than elements. We are, for the continued development process, interested in testing how snippets of code and logic for parameter mapping can blend with the current hierarchical structure for audio routing in WebAudioXML. Before building a prototype we set out a few design rules to steer the development:

Platform:
The syntax shall integrate with and become a part of WebAudioXML

Language:
The syntax shall use XML to declare variables and mappings using elements and attributes

Flexibility:
The syntax shall allow for flexible mapping and conversion of values

Readability:
The syntax shall be readable and comprehensible with focus on hierarchical structures and dependencies

1.4 The spreadsheet approach

An important influence for the current project is the approach used in spreadsheet applications like Excel. We value the affordances of spreadsheet applications for non-programmers to create logic and visual representation of data. The syntax proposed in this study borrows from that approach by offering XML elements and attributes to define variables and formulas in a similar way to data and formulas in a spreadsheet application. The difference is that the result of the calculation causes an audio parameter to change rather than a visual graph to update, see Fig. 1 and Listing. 1.

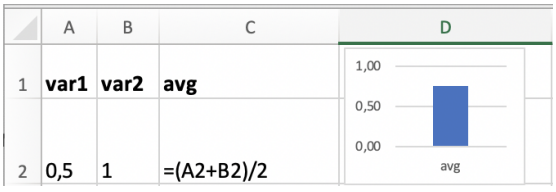


Figure 1. Spreadsheet formula and graph.

The example in Listing 1 shows how the new syntax proposed in this study uses XML elements and attributes to emulate a behaviour similar to the example from Excel in

[1] https://github.com/hanslindetorp/WebAudioXML/wiki/Parameter-Mapping

Fig. 1. The syntax is explained to a greater detail in Section 2.

```
<var name="var1" value="0.5"></var>
<var name="var2" value="1.0"></var>
<var name="avg" value="($var1+$var2)/2"></var>

<OscillatorNode>
  <frequency
    value="$avg"
    convert="Math.pow(10,x*3)">
  </frequency>
</OscillatorNode>
```

Listing 1. Code example for the Spreadsheet approach.

2. DEVELOPMENT PROCESS

Our aim with this study, in addition to the software development, is to gain more knowledge about strengths and weaknesses of the proposed syntax. We defined three cases as a starting point for the new specification and used an annotated portfolio [20] to collect insights from the design and development process. We also invited three experts from the field of sound and music computing to discuss the concept during the development process. Two participants are PhD students and one is a Postdoc, and all have extensive experience of using audio programming languages like Pure Data and SuperCollider in their artistic professions, and also in teaching the platforms at universities. They were interviewed individually and gave their consent for us to video record, analyse and use the results for research purposes. The interviews were between 30 and 60 minutes long where the participants contributed with both personal reflections and answers to prepared questions regarding clarity, strengths, weaknesses, potential, and challenges for the proposed syntax.

The three cases used as a starting point for this study spring from artistic sonification ideas formulated in our current student projects and act as a requirement specification for the syntax development. Below is a description of the three cases including the proposed syntax to solve them.

2.1 Case 1 – The Meta Knob: One-to-Many

In the first case, we identified a need for a complex audio configuration to be controlled on a meta level in a similar way as synthesizers can have a single knob for controlling multiple parameters at the same time. This makes it straightforward to build instrument plugins that can be used in sonification applications by developers without knowledge of how the instrument operates on a low-level. The first case requests variable objects anywhere in the XML-structure that can be read by multiple audio parameters on a lower level, see Fig. 2.

The <var>-element is similar to a cell in a spreadsheet-application. It can contain anything from simple data to complex formulas and mapping definitions for calculating and converting its value according to external variables (e.g. user interaction data) or other <var>-elements.

In Listing 2, the <var>-element named “param” continuously follows “$relX” which is a global variable referring

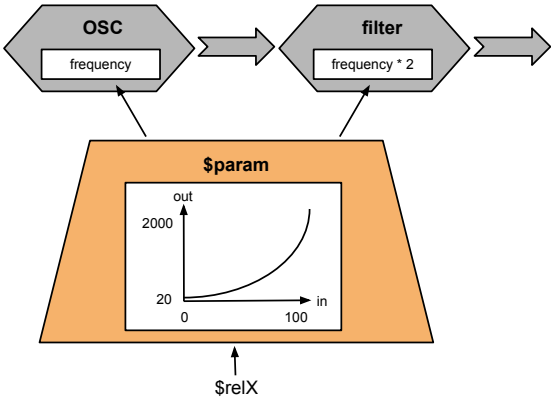


Figure 2. Systematic sketch for Case 1 with one-to-many mapping.

to the current horizontal position of the pointer. “$relX” is mapped exponentially from a value between 0 and 100 to a value between 20 and 2000 before it is stored in the variable object “$param”. Finally, the Oscillator and BiquadFilter nodes continuously update their frequency using the value of “$param” where the filter follows the same frequency value as the oscillator but one octave above.

```
<Chain>
  <var name="param"
    value="$relX"
    mapIn="0, 100"
    mapOut="20, 2000"
    curve="exp">
  </var>

  <OscillatorNode frequency="$param">
  </OscillatorNode>

  <BiquadFilterNode frequency="$param*2">
  </BiquadFilterNode>
</Chain>
```

Listing 2. Code example for Case 1 (see Fig. 2).

2.2 Case 2 – The Flexible Scale: Many-to-One

A common strategy for mapping values to frequencies is to quantize them to a musical scale, which is demonstrated in the second case. In traditional music though, the scale is rarely used without variation, and alterations typically occur depending on the direction of a phrase or the current harmony in the arrangement. The second case thus requests a solution where the state of multiple variables affect the frequency of an oscillator.

This case requires a new feature where an attribute of a <var>-element can follow the value of another <var>-element or external variable. Listing 3 illustrates how the pointer direction on the X-axis (“$dirX”) is mapped to the <var>-element “$third” causing it to have a value of either 3 or 4. This value is used in the <var>-element “$pitch” and refers to the number of semitones above the root note specified by the attribute “steps”. The variable “$pitch” is here continuously following the pointers position on the X-axis (“$relX”) and is mapped from values between 0 and

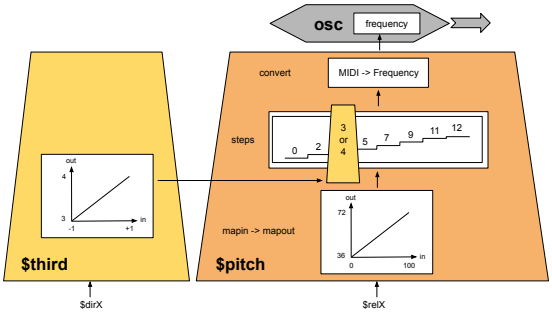


Figure 3. Systematic sketch for Case 2 with many-to-one mapping.

100 to a value between 36 to 72. After being quantized to a scale step (dynamically updated by "$dirX"), it is converted to a frequency in Hertz before the value is used to set the frequency of the OscillatorNode.

```
<var name="third" value="$dirX"
  mapin="-1, 1" mapout="3, 4">
</var>

<var name="pitch" value="$relX"
  mapin="0, 100" mapout="36, 72"
  steps="0, 2, $third, 5, 7, 8, 10, 12"
  convert="MIDI->frequency">
</var>

<OscillatorNode type="sine" frequency="$pitch">
</OscillatorNode>
```

Listing 3. Code example for Case 2 (see Fig. 3).

2.3 Case 3 – Complex Interpolation

In the third case we look for a non-limited way of defining interpolation curves for mapping an input value to a destination value. The metaphor we apply is the "automation"-feature typically found Digital Audio Workstations and graphical animation tools. Typically, this feature would map a time position to an audio or graphical parameter value using defined interpolation curves, but it could as well be used to describe any non-linear relationship between a variable and an audio parameter. To make this possible, the third case requires a syntax which supports multiple values for incoming values, outgoing values, curve shapes, steps and conversion functions.

While this case opens up for very complex mapping configurations, the addition to the WebAudioXML is fairly simple. The new feature supports multiple values for all attributes of a `<var>`-element or an audio parameter, see Listing 4 including "mapin", "mapout", "curve", "convert" and "steps". They can all be specified as one or several comma separated values with a few exceptions and restrictions: There has to be at least two "mapin"-values and typically the same number of "mapout"-values. More "mapout"-values will be ignored and fewer will be repeated. The number of "curve" and "convert"-values should be either a single or one less than the number of "mapin"-values to specify the interpolation between two values. If there are fewer "curve" or "convert"-value than

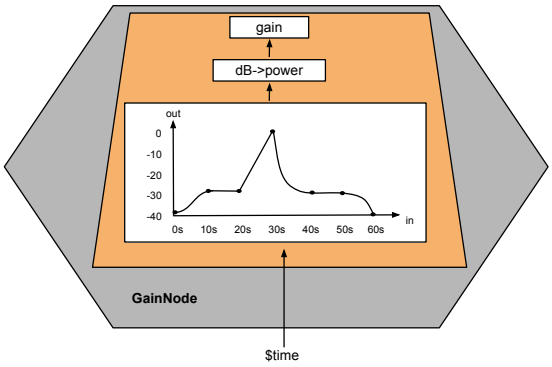


Figure 4. Systematic sketch for Case 3 with interpolation curves.

"mapin"-values, they will be repeated and used for multiple interpolation ranges.

In addition to presets such as "lin" and "exp", curves can also be specified as a mathematical expression in javascript like `Math.pow(x, 2)` where "x" is the mapped value scaled to a range of 0–1. There is also an extensive list with preset curves [2] with more intricate shapes including "easeIn", "easeInOut", "easeInOutQuad" etc. inherited from animation tools. These presets typically mimic behaviors in the physical world and can make a parameter interpolate from a linear variable in a way that e.g. accelerates in the lower part and retards in the upper. [3] While the curve-attribute affects the value before it is mapped to the mapout-value, the "convert"-attribute is the last step before the value affects the audio parameter. It can, similar to the curve-attribute take any javascript-expression to convert the value between different domains and also offers presets like "MIDI->frequency" and "dB->power".

The following code illustrates how the gain parameter can be controlled over time using different values and interpolation curves. The "$time"-variable refers to a user variable controlled from an external timer or slider using the javascript API `webAudioXML.setVariable("time", x)`.

```
<GainNode>
  <gain value="$time"
    mapin="0, 10, 20, 30, 40, 50, 60"
    mapout="-40, -30, 30, 0, -30, -30, -40"
    curve="easeInOut,lin,lin,easeOut,lin,exp"
    convert="dB->power">
  </gain>
</GainNode>
```

Listing 4. Code example for Case 3 (see Fig. 4).

3. RESULTS AND IMPLICATIONS

We present the result organized according to the design goals mentioned in Section 1.3. We also point out possible implications for further development related to each design goal.

[2] https://github.com/hanslindetorp/WebAudioXML/wiki/Parameter-Mapping

[3] https://easings.net/

3.1 Platform

The syntax shall integrate with and becoming a part of WebAudioXML.

The first proper tests of the new features was successful and proved the new syntax to be robust and straightforward to use with the WebAudioXML parser. The participants responded very positively to the potential on how Web Audio in general and WebAudioXML in particular can make audio applications easy accessible. One of them expressed that "it passes the grandma-test" when he realized that he could send someone a URL to a project and that it would run on any device without any installations. They also pointed out that the low entry style without the need for external classes makes it attractive for didactic and pedagogical purposes.

3.2 Language

The syntax shall use XML to declare variables and mappings using elements and attributes.

The first observation is that the new features merged into the previous syntax easily, while the implementation of them into the WebAudioXML-parser exposed some limitations and mistakes in the code design. The participants' response to the choice of XML for programming audio applications varied from being slightly surprised to heavily questioning. One participant objected to the use of XML for anything else but storing data and argued that the language loses its main purpose ("*the only thing XML is good for*") when the data is written using attributes with long strings rather than using the basic element tags. On the other hand, they agreed on that the hierarchical structure in XML is easy to understand as long as the configuration is limited to the constrains it provides.

The critique is interesting and raises the question about who the potential audience for a programming language is. WebAudioXML is primarily targeted towards web developers and should by design appeal more to that audience than to experts from the sound and music community. The critique regarding using XML for anything else than storing data is important and questions where the line should be drawn between storing data and logic. In this study we have explored the potential of implementing parts of the logic into the audio configuration model and to use the same language and file format for both. It promotes the building of reusable blocks in a similar way to that HTML has taken with custom elements but might be further discussed in terms of usability. The discussion also points towards a bigger question about what an application is and how it should be built. Similar to a spreadsheet application, WebAudioXML might not be the platform for a commercial products but rather offers new ways for prototyping ideas.

3.3 Flexibility

The syntax shall allow for flexible mapping and conversion of values.

All challenges from the three cases were solved and tested successfully. The new features also contributed to an update of old features like the possibility to write formulas directly into an attribute of an audio-element. The participants recognized several similarities with the proposed solution for parameter mapping to objects in other audio programming environments. UGens in SuperCollider and abstractions in PureData both offer similar functionality with inlets, outlets and a way of processing the value in-between. The experts perceived the platform as relatively limited in terms of possibilities and suggested a way of extending the format with a plugin-structure for adding any user-created audio object into the configuration.

Another idea that came up as a result of discussing flexibility was to let small blocks of data (e.g., a series of numbers specifying a musical scale) to be stored in a separate file. This would arguably make the code easier to read and the data more reusable and might work extra well for Case 3 above where the amount of data potentially could grow large. One of the participants also mentioned support for the Open Sound Control protocol [21] to make the platform even more flexible.

3.4 Readability

The syntax shall be readable and comprehensible with focus on hierarchical structures and dependencies.

The participants were in general very appreciative regarding the accessibility and the descriptive language in WebAudioXML. They pointed out that the proposed naming conventions and coding strategies were clear and easy to understand. This being said, they were more skeptical regarding the use of long strings to express complex mappings. They argued that it shared similarities with old music programming languages like CSound and that the long strings used in attributes might be a bit scary for many developers. They also asked for contextual documentation as they would expect from a coding environment. While they expressed appreciation for the clear, hierarchical structure and relations between objects, parameters, and variables, they also pointed out the potential problem when audio signals and parameter mappings are set to break those structures and suggested a graphical interface to get a better overview.

We find the discussion about a graphical interface especially intriguing; it is easy to lose flexibility in favor for the usability a graphical interface can offer. One possible compromise that both stays true to the text-based concept of XML and offers contextual guidance in some XML editors is the use of a well-designed XSD-file. [4] One feature that was discussed by all participants was case 3 and the complex relation between multiple input and output values. Even if they thought the syntax was clear, all of them mentioned that the feature was hard to visualize without a graphical tool available.

[4] `https://www.w3.org/XML/Schema`

3.5 Other observations

The annotated portfolio from the development process also contributes to important insights. It was obvious that the addition of the new features to the parser challenged several design decisions made in the beginning of the WebAudioXML development. Some of them were updated during this process but there is still a need for a few new class definitions. The parser also contains some non-recommended solutions like `eval(expression)` that work well for a prototype but is a potential security risk if put into production. Furthermore, the "value"-attribute replaced the old "follow"-attribute. Its approach with any type of expression from a single target variable to a complex mathematical expression containing multiple variables proved to be a more flexible way to receive data.

One feature we struggled with during the implementation was the sequence of the mapping steps inside a variable object. Even if the attributes themselves can be defined in any order in the XML-element, we decided to make them always operate in the following order, and if any attribute is missing, that step will be bypassed.

```
value->mapin->curve->mapout->steps->convert
```

We also got new ideas for further development. One is to implement the structure used in case 2 for all attributes in a variable object, including mapout, curve and values in a convert expression. Another is to add built-in features for statistical evaluation in a variable and map the data according to the result. This could e.g. make it easy to send peak values to an audio parameter when the incoming data is near the confidence limits of a normally distributed dataset.

Finally, we think that it would be a great addition if the variable object could have built-in support for derivative and second order derivative which would be useful for applications aiming for physical interaction.

4. CONCLUSION AND FUTURE DIRECTION

The proposed syntax for audio parameter mapping has been designed, discussed and implemented into the latest version of WebAudioXML. The participating experts pointed at several promising features of the development. First, they confirm that the proposed solutions match the design goals that were set out. Second, they confirm the didactic and pedagogical value and potential of the WebAudioXML framework. Third, we noticed that attitudes towards XML would influence their immediate understanding, both positively and negatively. It is quite clear that the experts don't regard themselves as primary users of the framework, which is both expected and intentional. As such, the ambition of creating this system and its scope have been met.

We were particularly encouraged by the participants to be consequent in naming variables and to find ways for contextual help with syntax explanation and suggestions. The suggested need for and solution to case 3 was generally more difficult for them to grasp than cases 1 and 2, and would require a graphical interface to become useful. It was stated that syntax and concepts borrowed from other languages were easy to understand and that the concept of external files with top-level parameters exposed to the parent object was an expected feature relating to object oriented programming in general and "abstractions" in Pure Data in particular.

Forthcoming studies involving students in our courses will target parameter mappings in particular. Then, both the pedagogical potential and the syntax itself will be systematically evaluated with regards to what the students choose to create and how they will use WebAudioXML to accomplish their artistic goals.

Acknowledgments

Thanks to the participants for contributing with highly valued insights and feedback.

5. REFERENCES

[1] B. J. Grond F., "Chapter 15: Parameter mapping sonification," in *The Sonification Handbook*, 2011, p. 366.

[2] M. A. Nees and B. N. Walker, "Listener, task, and auditory graph: Toward a conceptual model of auditory graph comprehension," in *International Conference on Auditory Display*. Georgia Institute of Technology, 2007.

[3] J. G. Neuhoff, J. Wayand, and G. Kramer, "Pitch and loudness interact in auditory displays: Can the data get lost in the map?" *Journal of Experimental Psychology: Applied*, vol. 8, no. 1, p. 17, 2002.

[4] J. Rohrhuber, "S–introducing sonification variables," in *In Proceedings of the Supercollider Symposium*, 2010.

[5] B. Vercoe, *Csound*, 1985 (accessed May 28, 2021). [Online]. Available: https://csound.com/

[6] M. Puckette, "Max at seventeen," *Computer Music Journal*, vol. 26, no. 4, pp. 31–43, 2002.

[7] M. S. Puckette, "Pure data," in *Proceedings: International Computer Music Conference 1997, Thessaloniki, Hellas, 25-30 september 1997*. The International Computer Music Association, 1997, pp. 224–227.

[8] J. McCartney, "Rethinking the computer music language: Supercollider," *Computer Music Journal*, vol. 26, no. 4, pp. 61–68, 2002.

[9] G. Wang, P. R. Cook *et al.*, "Chuck: A concurrent, on-the-fly, audio programming language," in *ICMC*, 2003.

[10] R. M. Candey, A. M. Schertenleib, and W. Diaz Merced, "Xsonify sonification tool for space physics," in *International Conference on Auditory Display*. Georgia Institute of Technology, 2006.

[11] S. Pauletto and A. Hunt, "A toolkit for interactive sonification," in *International Conference on Auditory Display*. Georgia Institute of Technology, 2004.

[12] A. Schoon and F. Dombois, "Sonification in music," in *International Conference on Auditory Display*. Georgia Institute of Technology, 2009.

[13] D. Worrall, M. Bylstra, S. Barrass, and R. Dean, "Sonipy: The design of an extendable software framework for sonification research and auditory display," in *Proc. ICAD*, 2007.

[14] H. Lindetorp and K. Falkenberg, "WebAudioXML: Proposing a new standard for structuring web audio," in *Sound and Music Computing Conference*. Zenodo, 2020 (accessed May 28, 2021), pp. 25–31, qC 20200722. [Online]. Available: https://zenodo.org/record/3898655#.X3HgbC0zLa4

[15] H. Lindetorp, *iMusic: JavaScript framework for interactive music*, 2016 (accessed May 28, 2021). [Online]. Available: https://github.com/hanslindetorp/imusic

[16] ——, *WebAudioXML Sonification Toolkit*, 2020 (accessed May 28, 2021). [Online]. Available: https://github.com/hanslindetorp/SonificationToolkit

[17] K. F. Hansen, R. Bresin, A. Holzapfel, S. Pauletto, T. Gulz, H. Lindetorp, O. Misgeld, and M. Sköld, "Student involvement in sound and music research: Current practices at KTH and KMH," in *Proceedings of the first Nordic SMC*. Zenodo, 2019, pp. 36–41.

[18] H. Lindetorp, "Immersive and interactive music for everyone," in *Proceedings of the Nordic Sound and Music Computing Conference 2019 (NSMC2019) and the Interactive Sonification Workshop 2019 (ISON2019) :*, 2019, pp. 16–20. [Online]. Available: http://smcsweden.se/proceedings/NordicSMC_ISon_2019_Proceedings.pdf

[19] P. Adenot and R. Toy, *Web Audio API: W3C Candidate Recommendation, 18 September 2018*, 2018 (accessed May 28, 2021). [Online]. Available: https://www.w3.org/TR/2018/CR-webaudio-20180918/

[20] B. Gaver and J. Bowers, "Annotated portfolios," *interactions*, vol. 19, no. 4, pp. 40–49, 2012.

[21] A. Freed, "Open sound control: A new protocol for communicating with sound synthesizers," in *International Computer Music Conference (ICMC)*, 1997.

THE GRAMOPHONE: A PROGRAMMABLE DMI TO FACILITATE THE TEACHING OF STEM DISCIPLINES

Romain MICHON (michon@grame.fr) [1,2], **Catinca DUMITRASCU**[1], **Stéphane LETZ**[1], **Yann ORLAREY**[1], and **Dominique FOBER**[1]

[1]**GRAME, Centre National de Création Musicale**, Lyon, France
[2]*Center for Computer Research in Music and Acoustics (CCRMA)*, **Stanford University**, CA USA

ABSTRACT

In this paper, we introduce the Gramophone, a Digital Musical Instrument programmable with the FAUST programming language. The Gramophone has been designed as part of the Amstramgrame project which aims at facilitating the teaching of STEM disciplines through audio programming. The Gramophone is completely standalone and it hosts a wide range of sensors to control sound synthesis parameters. Its programming is carried out through a pedagogical web platform. A total of 160 Gramophones were produced as part of Amstramgrame and are now touring in French public schools. This paper presents the design process that lead to this instrument, its implementation and evaluation, as well as future directions for this work.

1. INTRODUCTION

STEAM [1] projects have become popular in recent years, taking different approaches combining the teaching of Sciences with the Arts. The field of music technology has been acting as a natural platform for the development of this type of project [1–4]. *iMuSciCA* [5] is very representative by aiming at facilitating the teaching of sciences through music technology via a Web platform [6] where students can design their own musical instruments using physics and mathematics principles, visualize the physical parameters of a sound in the prospect of making a composition, etc.

While most STEAM projects in the field of music technology focus on the combination of Sciences with the Arts, fewer emphasize computer programming. In that context, we recently launched the Amstramgrame project [2] [7] which aims at facilitating the teaching of maths and physics through the programming of a physical Digital Musical Instrument (DMI): the Gramophone (see Figure 1).

The Amstramgrame website plays a central role serving both as:

Figure 1. The Gramophone.

- a pedagogical platform for teachers and students providing pre-written lessons,
- an environment to program the Gramophone.

A typical Amstramgrame module begins with a maths or physics teacher, leads to the development of an audio DSP algorithm that in turn, is used to program the Gramophone to make a custom musical instrument. Finally, students work with a composer to improve their instrument and to write a piece that is performed by the class during a final performance.

The Gramophone was specifically developed for Amstramgrame to provide a platform to make the programming of audio DSP [3] algorithms (and hence, of mathematical concepts studied in class) more tangible. It is lightweight, completely standalone (it hosts a built-in rechargeable lithium battery and a speaker), robust, and it can be programmed using the FAUST programming language [4] [8] through a web platform. A 9 DoF motion sensor as well as a physical button and rotary potentiometers can be used to control the parameters of the synthesized sound. The Gramophone was designed to be held in one hand and can be easily moved around by the performer thanks to its strap (see Figure 1).

In this paper, we first present the early prototypes of the Gramophone that lead to its final design. We then describe

[1] *Science, Technology, Engineering, Arts, and Mathematics*
[2] https://www.amstramgrame.fr/

[3] *Digital Signal Processing*
[4] https://faust.grame.fr

the hardware and software implementation of this instrument before providing information on its diffusion as well as the results of an evaluation conducted on about 180 students and over 160 gramophones. We finally reflect upon future directions for this project.

2. EARLY PROTOTYPES

The development of the Gramophone was initiated in January 2019. We wanted this instrument to meet a certain number of requirements by being:

- standalone (e.g., battery powered and host a built-in speaker),
- programmable with FAUST,
- playable with one hand,
- quick at booting,
- robust and durable (and hence adapted to primary and middle school students).

Given the size constraints, the immediate boot time, and portability requirements, we decided to build this instrument around a microcontroller powerful enough to carry out real-time audio DSP. The best candidate at the time was the Teensy 3.6, [5] which provided unparalleled computational power (180 MHz) and support for real-time audio processing through the Teensy audio library [6] as well as an "audio shield" [7] (audio codec sister board). In that context, we implemented `faust2teensy` [9], a command line tool allowing for the programming of the Teensy with FAUST for real-time audio DSP applications.

Our initial prototype (see Figure 2) was designed to be held as a stick and looked a little bit like a rattle (or a pan). The only continuous controller was a 9 DoF motion sensor as well as a button mounted on the handle of the instrument. Its shape potentially limited the orientation that could be given to the object and mobilized all the fingers of the hand to hold the device.

The second iteration of the Gramophone (see Figure 3) solved this issue and gave access to a joystick that could be controlled by the thumb of the hand holding the instrument. A photoresistor was also added to the handle.

Both designs implied the use of an external battery charging circuit ("battery boost") connected to the Teensy as well as a small amplifier for the speaker. The battery boost tended to create ground issues with the Teensy resulting into background noise. All in all, we hoped for a all-in-one solution where the microcontroller, the motion sensor, the amplifier, and the battery charging circuit could all be integrated in the same board. This was a big concern since we were planning on making a fairly large number of Gramophones for Amstramgrame (we were thinking of at least 100 at the time), so the design had to be as optimized as possible.

[5] `https://www.pjrc.com/store/teensy36.html`

[6] `https://www.pjrc.com/teensy/td_libs_Audio.html`

[7] `https://www.pjrc.com/store/teensy3_audio.html`

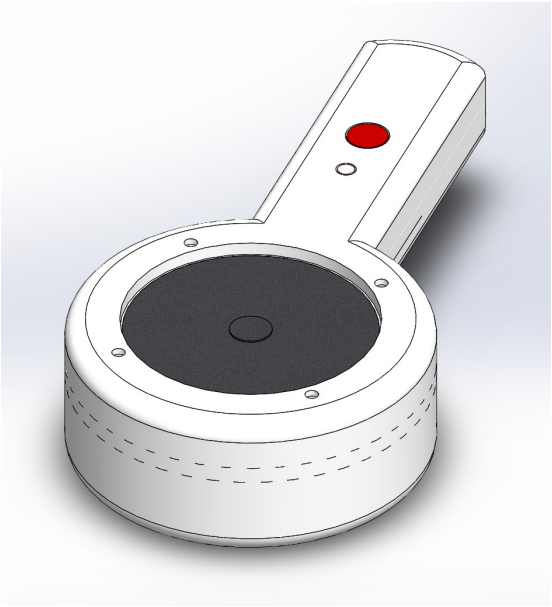


Figure 2. The initial prototype of the Gramophone.

3. THE GRAMOPHONE

3.1 Overview

Only a few months after the first Gramophone prototypes were completed, LilyGO released a new ESP-32 based board: the TTGO TAudio. [8] For only $15, this board hosts an ESP-32 microcontroller (with built-in WiFi and Bluetooth and a dual core processor operating at 240 MHz), a 9 DoF motion sensor, a battery charging circuit, and an audio codec with a built-in amplifier. We therefore saw it as a way to solve many technical problems that we had with using the Teensy to make our instrument. In that context, we developed `faust2esp32` [10], a command line tool to program ESP32 based boards with FAUST, making this language the only alternative to C++ to program this platform.

The final version of the Gramophone is built around a LilyGO TTGO TAudio board (see Figure 4 for the initial prototype of the final version, Figure 1 for the final distributed version, and Figure 6 for a detailed view of the different features of the Gramophone). It fits in one hand and can be easily moved around thanks to its strap. A photoresistor lands directly under the thumb of the performer. Similarly, a button finds its place under the middle finger. A rotary potentiometer can be controlled with the left hand of the performer. The Gramophone benefits directly from the built-in 9 DoF sensor of the TTGO TAudio board. All these sensors can be assigned to sound synthesis parameters directly from the FAUST program (see §4).

The Gramophone is charged and programmed through a micro USB cable which connects to the corresponding port on the TTGO TAudio board. A 2.05 Ah lithium battery ensures an extended autonomy for the Gramophone which

[8] `http://www.lilygo.cn/`

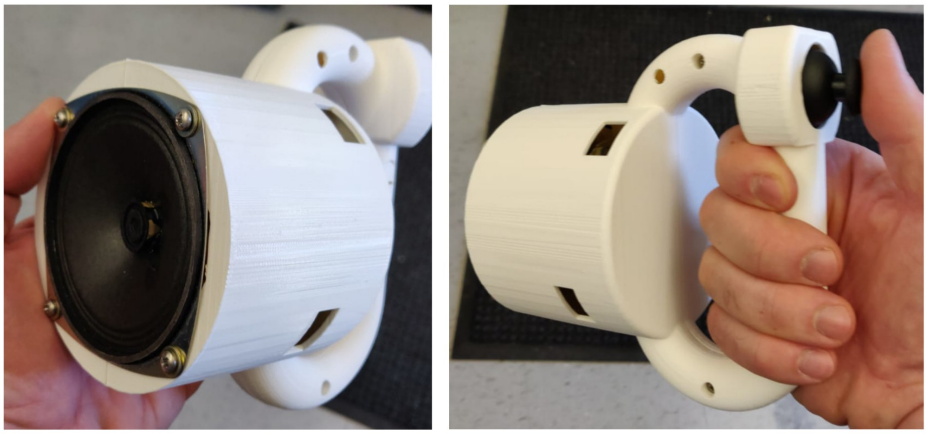


Figure 3. The second prototype of the Gramophone.

Figure 4. The third prototype of the Gramophone.

can be played for about 24 hours in battery mode (with a standard use). A microphone can be used to process sound in real-time. A dedicated rotary potentiometer can be used to control the volume of the instrument. Finally, a rotary encoder allows the performer to cycle around different FAUST programs installed on the Gramophone.

3.2 The Gramophone Case

The Gramophone case was entirely designed in OpenScad [9] and Inkscape [10] (through Inkscape2scad [11]). Since we never intended to sell the Gramophone, we wanted to make sure that it could easily be reproduced (and/or modified) using tools accessible to everyone. The source files of the Gramophone case as well as the corresponding STL files can be found on the Amstramgrame GitHub. [12] Since a large number of Gramophones had to be made for Amstramgrame (over 160), the 3D printing of their case was outsourced.

3.3 Electronics and Assembly

Beside its TTGO TAudio board, the Gramophone relies on fairly simple electronic components (i.e., speaker, battery, potentiometers, buttons, etc.). An exhaustive video tutorial on how to assemble a Gramophone can be found on the Amstramgrame website. [13] Assembling a Gramophone from scratch takes about three hours. To make the 160 Gramophones of Amstramgrame, we hired an "assembler" for a period of five months whose role was also to test each Gramophone before shipping.

4. SOFTWARE ENVIRONMENT

4.1 Programming Environment

The Amstramgrame website contains dozens of Gramophone program examples spread across a wide range of tutorials. Each example can be edited in a simplified version of the FAUST Web IDE [11] specifically designed as part of Amstramgrame. [14] The sound corresponding to a Gramophone FAUST program can be heard and prototyped directly in the web browser. A "Gramo" button can be pressed to generate the firmware corresponding to the FAUST program ready-to-be-installed on the Gramophone.

The GramoLoader [15] (see Figure 8) is a software tool designed as part of Amstramgrame acting as a bridge between the FAUST Web IDE and the Gramophone. It takes the firmware generated when pressing the "Gramo" button and installs it on the Gramophone. Once the firmware has been selected once, it will be automatically installed

[9] https://www.openscad.org/
[10] https://inkscape.org/
[11] https://github.com/grame-cncm/faust/tree/master-dev/tools/physicalModeling/inkscape2scad
[12] https://github.com/amstramgrame/amstramgrame
[13] https://www.amstramgrame.fr/en/gramophone/making/
[14] https://faustide.grame.fr/?mode=amstram
[15] https://www.amstramgrame.fr/gramophone/loader/

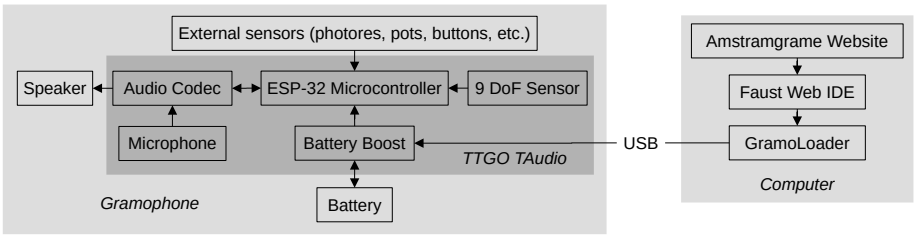


Figure 5. Overview of the components of the Gramophone.

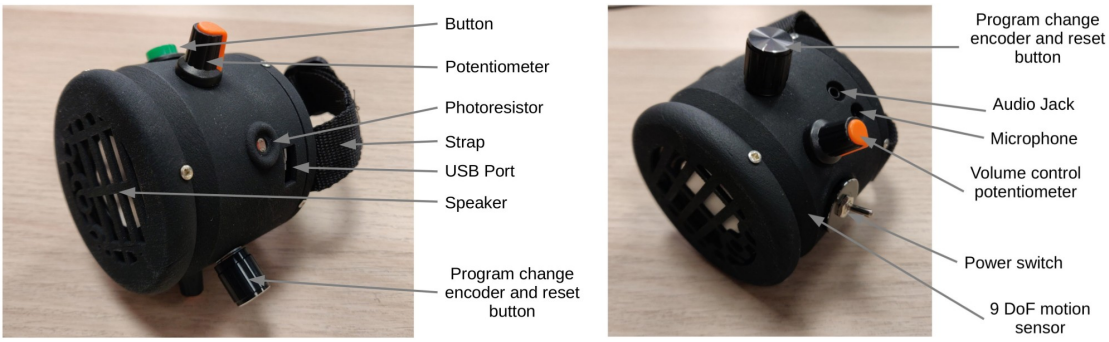


Figure 6. The various elements of the Gramophone.

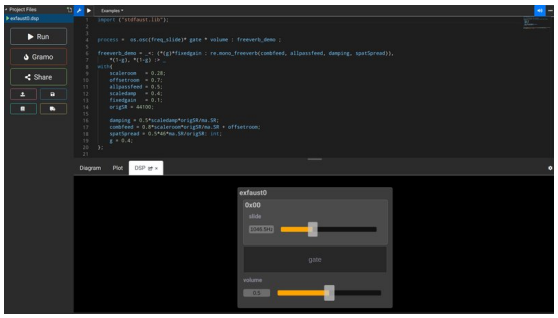


Figure 7. The Amstramgrame FAUST Web IDE.

Figure 8. The GramoLoader.

on the Gramophone every time the "Gramo" button will be pressed. The GramoLoader is written in Python, is available for Windows, MacOS, and Linux, and it is completely open source.

4.2 Programming the Gramophone

4.2.1 Mapping Controllers to DSP Parameters

A Gramophone program can be as simple as:

```
import("stdfaust.lib");
f = hslider("freq[knob:2]",
    440,50,1000,0.01) : si.smoo;
t = button("gate[switch:1]") : si.smoo;
process = os.osc(f)*gate;
```

In that case, the frequency of a sine wave oscillator is controlled by the rotary potentiometer of the Gramophone and sound can be triggered by pressing the button under the middle finger. In other words, any FAUST parameter (i.e., declared with a slider, a `nentry`, a `button`, etc.) can be mapped to a Gramophone controller using a metadata. Standard FAUST motion sensors (i.e., gyroscope, accelerometer, etc.) metadatas can be used to map the 9 DoF sensor of the Gramophone to FAUST parameters. An exhaustive list of the Gramophone FAUST metadata can be found on the Amstramgrame website. [16]

4.2.2 MIDI Bluetooth

MIDI Bluetooth support has recently been added to the Gramophone. A Bluetooth device name can be declared using the `btmidi_device_name` general metadata. Standard FAUST MIDI metadatas [17] can then be used.

[16] `https://www.amstramgrame.fr/gramophone/about/#metadatas-de-programmation-du-gramophone`
[17] `https://faustdoc.grame.fr/manual/midi/`

In the following examples:

```
declare btmidi_device_name "Gramophone";
f = hslider("freq[midi: ctrl 0]",
    440,50,1000,0.01) : si.smoo;
t = button("gate[switch:1]") : si.smoo;
process = os.osc(f)*gate;
```

the frequency parameter of the sine wave oscillator is controlled using MIDI CC 0 and sound is triggered by pressing the button on the Gramophone.

In the previous example, Bluetooth pairing has to be initiated by an external device (e.g., a smartphone, a computer, etc.) but the Gramophone itself can be configured to initiate pairing by specifying the name of the target with the `btmidi_remote_name` general metadata. E.g.,:

```
declare btmidi_remote_name "External MIDI
    Keyboard";
```

This also can be used to connect two Gramophones together to potentially synchronize them, etc.

4.2.3 Limitations

The programming limitations of the Gramophone are directly linked to that of the TTGO TAudio board [10]. For instance, the limited amount of RAM (about 1MB) has a significant impact on the kind of DSP algorithms that can be ran on the Gramophone. For instance, long delays and echos or complex reverbs will likely not work on this instrument. The same is true for table-based oscillators, but the FAUST compiler now integrates a system to automatically adapt table sizes when using embedded systems requiring small memory footprints.

Computational power is less of an issue and the dual core processor of the TTGO TAudio board running at 240 MHz is powerful enough to run a wide range of DSP algorithms.

5. DIFFUSION AND EVALUATION

Over 160 Gramophones were produced as part of Amstramgrame to constitute "Gramophone backpacks" containing 30 Gramophones each. A "Gramophone assembler" was hired for a period of five months to carry out this task (see §3.3). All Gramophones are owned by the Amstramgrame project: we currently do not sell them since this would be too complicated for us from a legal standpoint.

The pilot phase of Amstramgrame (see Figure 9) ended in March 2021 and was carried out in two different schools, targeting a total of eight classes (middle and high school level) corresponding to about 180 students for a total of eighty hours of Amstramgrame sessions [7]. From a technical standpoint, the Gramophones and their tool-chain proved to be relatively stable. Only 4 Gramophones broke during this period (and were immediately and easily fixed). The web platform never stopped working and both students and instructors made positive comments about all the tools used as part of Amstramgrame. Gramophones ended up being adapted to most hands (large or small) and having satisfying ergonomics. A large number of users highlighted their good sound quality and loudness.

Figure 9. An Amstramgrame session at the *Cité scolaire du Cheylard* (Ardèche, France) – with social distancing.

Now that the pilot phase of the project is over, Amstramgrame will be implemented at a larger scale in a wide range of schools of the Auvergne-Rhône-Alpes region in France. The scope of the project had to be slightly limited/adapted due to the Covid-19 pandemic though.

6. FUTURE WORK

At this point, we consider that the Gramophone reached a certain level of stability and we don't plan to make any modifications to it in the future. However we do believe that its programming tool-chain could be improved. For instance, we would like to get rid of the GramoLoader to replace it with a system fully integrated in the Web browser (a bit like the Arduino Create Platform [18]). Indeed, the fact that the GramoLoader is platform-dependent and requires an installation can be a significant problem in schools that don't have a permanent system administrator (which is often the case in France). Having a comprehensive web solution would solve this problem and make Amstramgrame more portable.

We're currently working on an adapted version of the FAUST Playground [19] allowing for the programming of the Gramophone. Unlike in the FAUST Web IDE, FAUST programs can be assembled using a graphical patching environment in the FAUST Playground, making it more accessible to younger students.

Our ambition is to add a hardware prototyping aspect to Amstramgrame by offering the possibility to substitute Gramophones with kits allowing students to make their own instrument using sensors and cardboard (in a similar way as Nitendo labo [20]). The programming tool-chain would remain the same as for the Gramophones. The development of these kits was recently initiated.

7. CONCLUSIONS

Amstramgrame gave us the opportunity to (i) develop a DMI from scratch: the Gramophone, (ii) produce it at a fairly large scale, and (iii) evaluate its use by a large group

[18] `https://create.arduino.cc/editor`
[19] `https://faustplayground.grame.fr/`
[20] `https://labo.nintendo.com/`

of people. We believe that it fulfilled its initial objective which was to provide a platform to make more tangible the various scientific concepts approached in class in the context of Amstramgrame.

While the use of the Gramophones has been limited so far to pedagogical purposes, we plan to take advantage of the fact that we own a large number them to use this instrument in the context of professional music productions to create a "Gramophone orchestra."

Acknowledgments

The development of the Gramophone and of its programming tool-chain has been carried out as part of the Amstramgrame project which is funded by the métropole de Lyon, the DRAC Auvergne-Rhône-Alpes, the SNCF foundation, and the Blaise Pascal foundation.

8. REFERENCES

[1] J. Shafer and J. Skripchuk, "Computational thinking in music: A data-driven general education steam course," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20)*, Portland (USA), 2020.

[2] J. Yoon and K. J. Kim, "Science song project: Integration of science, technology and music to learn science and process skills," *K-12 STEM Education*, vol. 3, pp. 235–250, 2017.

[3] J. Gregorio, D. S. Rosen, and B. G. Morton, "Introduction to steam through music technology (evaluation)," in *Proceedings of the 122nd ASEE Annual Conference and Exposition*, Seattle (USA), 2015.

[4] J. Dorfman, "Exploring models of technology integration into music teacher preparation programs." *Visions of Research in Music Education*, vol. 28, 2016.

[5] V. Katsouros, E. Fotinea, R. Frans, E. Andreotti, P. Stergiopoulos, M. Chaniotakis, T. Fischer, R. Piechaud, Z. Karpati, P. Laborde, D. Martín-Albo, F. Simistira, and M. Liwicki, *iMuSciCA: Interactive Music Science Collaborative Activities for STEAM Learning*. Cham: Springer International Publishing, 2018, pp. 123–154.

[6] K. Kritsis, M. Bouillon, D. Martín-Albo, C. Acosta, R. Piéchaud, and V. Katsouros, "imuscica: A web platform for science education through music activities," in *Proceedings of the Web Audio Conference WAC-2019*, Trondheim (Norway), 2019.

[7] R. Michon, C. Dumitrascu, S. Chudet, Y. Orlarey, S. Letz, and D. Fober, "Amstramgrame: Making scientific concepts more tangible through music technology at school," in *Proceedings of New Interfaces for Musical Expression Conference (NIME-21)*, Shanghai (China), 2021.

[8] Y. Orlarey, D. Fober, and S. Letz, *Faust: an Efficient Functional Approach to DSP Programming*. New Computational Paradigms for Computer Music. Delatour, 2009.

[9] R. Michon, Y. Orlarey, S. Letz, and D. Fober, "Real time audio digital signal processing with faust and the teensy," in *Proceedings of Sound and Music Computing Conference (SMC-19)*, Malaga (Spain), 2019.

[10] R. Michon, D. Overholt, S. Letz, Y. Orlarey, D. Fober, and C. Dumitrascu, "A faust architecture for the esp32 microcontroller," in *Proceedings of Sound and Music Computing Conference (SMC-20)*, Turin (Italy), 2020.

[11] S. Ren, S. Letz, Y. Orlarey, R. Michon, D. Fober, M. Buffa, E. Ammari, and J. Lebrun, "Faust online ide: dynamically compile and publish faust code as webaudio plugins," in *WAC 2019-5th Web Audio Conference*, 2019.

VIRTUAL ROBOTIC MUSICIANSHIP: CHALLENGES AND OPPORTUNITIES

Thales R. P. PESSANHA[1,2], **Higor CAMPOREZ**[4], **Jônatas MANZOLLI**[1,2], **Bruno S. MASIERO**[1,3], **Leandro COSTALONGA**[4], and **Tiago F. TAVARES**[1,3]

[1]*Interdisciplinary Nucleus for Sound Studies*, **University of Campinas**, Campinas, Brazil
[2]*Institute of Arts*, **University of Campinas**, Campinas, Brazil
[3]*School of Electrical and Computer Engineering*, **University of Campinas**, Campinas, Brazil
[4]*School of Computing and Electronics*, **Federal University of Espírito Santo**, São Mateus, Brazil

ABSTRACT

In the last few decades, robots have fostered unique possibilities for musical performance and composition, allowing novel interactions with musicians and memorable experiences for the audience. Robotic musicians can be built in many shapes and have diverse functionalities, making robot musicianship a fertile research field. However, building physical robots requires access to electrical and mechanical components, as well as laboratory equipment, which can make them financially unfeasible in peripheral countries. Moreover, building physical experimental devices quickly raises the problem of disposing of broken or outdated parts. Finally, the COVID-19 crisis has decreased access to laboratories and forced social isolation, which further harms physical robots' development. In this position paper, we argue that the current technology for robot simulation can be used to provide most aspects of physical robots, with considerable advantages related to the financial cost, the environmental impact, and the possibility of testing and sharing robots using the Internet. We also discuss previous work on virtual presence, which indicates that both the performers and the audience can feel being present in the same space as the virtual robots. Lastly, we anticipate challenges and research opportunities in this field of research.

1. INTRODUCTION

Robots are an important category of musical agents [1] and have been used in music-making for a few decades now [2–6]. Robotic musicians [7] use sensors, actuators, and electronic devices, bringing a diversity of new perspectives when compared to piano players and other mechanical actuators, like the visual cues that relate to their movements and sound qualities [8], and the possibility producing realistic sounds from physical movements [8].

Musical robots can also benefit from sensing their musical surroundings [9–11], which allows them to automatically respond to specific cues. Shimon [12], for instance, is a robotic marimba that is capable of interacting with a human musician in real-time. These autonomous responses can help mitigate the inevitable, even if short, actuator delays [13]. Bear in mind, that robotic musicians have mechanical delays caused by electromechanics components. Thus, these robots need to compensate them to keep in time. Dealing with delays and synchronization issues in robotic music performance is quite a challenge [14]. Also, autonomous musical cue anticipation can allow slower, more expressive gestures that allow human musicians to create expectations towards the robot's behavior [15].

However, robotic musicianship has some drawbacks. First, the cost of high-quality electronic parts in peripheral countries that can quickly make robot-making economically unfeasible [16]. Such a cost hinders the development of musical robots, and, consequently, of their surrounding elements: musical composition and performance, gesture and interaction design, and device building and control. Then there is the environmental impact of importing electronic goods from abroad and all the electronic waste that is produced during the process [17]. Finally, due to the social isolation enforced by the COVID-19 sanitary crisis, the access to the electronic laboratories and music studios has been compromised, as well as the possibility to collaborate in person with the multidisciplinary team that is required to build such robots.

In this paper, we propose using virtualization to enable robotic musicianship and their surrounding developments. For such, we use a simulator to devise robots and their behaviors. Then, musicians can interact with the robots in the same way as they would with hardware-based robots, which can be contradictory statement since robots, by definition, are indeed physical machines [18].

The idea of using virtual robots has been used in the past. In special, we note a marimba player designed within SolidWorks [19], and a virtual drummer that helps in music therapy for rehabilitation [20]. However, we are not aware of initiatives towards building virtual robot musicians for music production, research, and performance.

It is noticeable that recent technologies on physical simulation have allowed simulating realistic behavior that can be used either for games or for practicing real-life situations [21]. Also, recent developments in sound processing algorithms and hardware have allowed realistic sound synthesis and binaural spatialization. These conditions allow virtual robots to have a consistent behavior, which can lead to highly engaging and immersive experiences.

Virtual robots have the obvious advantage of overcoming economic costs, which is greatly relevant in peripheral countries. Also, they can be immediately replicated, allowing to create unlimited-size orchestras. They can be shared between developers and musicians in different locations, which greatly facilitates the interaction, in special during the pandemics and they have a smaller ecological footprint.

Following, we further discuss the challenges and opportunities related to virtual robotic musicianship.

2. PROBLEMS RELATED TO PHYSICAL ROBOTS

Non-human devices for music making have been part of the human imaginary since centuries-old legends like the singing harp of Jack and the Beanstalk. Real devices capable of playing music can be found in Leonardo DaVinci's sketches for mechanical drums, and, more recently, in the player pianos from the 19th Century. In the second half of the 20th Century, robots became increasingly present in a diversity of musical applications

Musical robots are now present in art, industry, academia, and in hobby projects, but their development is far from trivial. Building and using musical robots requires a series of skills that span between Engineering and Music, that is, the robot-building team must be skilled in a number of topic such as electronics, mechanics, programming microcontrollers, designing musical interactions, composing music, and playing music.

The development in electronics and mechanics requires special machinery, such as oscilloscopes or turnstiles, which can be inaccessible outside institutional laboratories. Also, each development cycle can require the acquisition of new electronic parts, which have a cost of their own. This makes the development cycle of physical robots considerably slow and expensive, in special in developing countries [16].

The need for a team and laboratory is a problem of its own during the COVID19 pandemic. In this context, social isolation calls for the temporary closing of working environments, and the impossibility of working together makes it unfeasible to join a team with all necessary skills.

Even in non-pandemic times, despite their expensive and time-consuming development, physical robots can quickly become outdated or malfunctioning, as any experimental electronic device. The constant development of musical robots generates clutter, which has an inherent problem related to the disposal of electronic components. Some parts of malfunctioning robots can be disassembled for usage in future experiments, and some robot components can be built from reclaimed hardware such as old printers or smartphones [22] [23] but others – especially malfunctioning electronics, soldered components, and 3D-printed parts – are simply discarded, which further contributes to already complicated environmental issues [17].

All of these difficulties can be overcome, as they have been, with an aim to bring to life the musical automatons that have long resided the human imaginary. However, we note that current technology allows simulating realistic physics and sound production, which has enabled, for example, virtual reality in games and other applications. This means that all of these difficulties can be mitigated by using virtual counterparts of physical robots, in a path that resembles the migration from physical synthesizers to virtual ones. Furthermore, we can have virtual (software) robots, which can be stored in hard drives, shared over the Internet, and built as a massive collaborative effort.

Next, we discuss the potential impacts of virtual robots in the context of musical performance.

3. PRESENCE AND VIRTUAL ROBOTS

Physical robots have been shown to provide a diversity of possibilities in musical creation and performance. Musicians and the audience are *present* in the same space as the robots. This section discusses the concept of Presence and the subsequent potential for using virtual robots in musical performance.

In literature, Presence is often referred to as telepresence, virtual Presence, or mediated Presence. In general, Presence can be understood as a psychological state or subjective perception in which an individual's experience mediated by technology, in part or whole, fails to accurately recognize the role of technology in his individual experience [24]. For Slater and Usoh [25], Presence can be defined as the virtual reality users' suspension of the disbelief that they are in a different world from where their bodies are physically located. We understand it is possible to use virtual robots in interactive music performances using the Presence concept. Thus, we foresee a virtual community implemented in mixed reality, inhabited by physical, remote, and virtual robots. This community raises the fundamental question of if such a virtual environment can generate and mediate real-time musical performances.

The term "Telepresence" was first defined by Marvin Minsky [26], who emphasized the possibility that humans could feel the sensation of being physically immersed or transported to a remote workspace through teleoperation systems. However, the definition coined by Minsky projected the development of high-quality simulation refinements and sensory feedback technologies. He predicted using telepresence in dangerous activities, the development of telemedicine technology, and the possibility of home office work – which anticipated the intense use of virtual technologies induced by the COVID19 pandemics. Further, Misky [26] defines social Presence as when users feel they are with other people locally or remotely, and copresence as when someone feels that they are co-located elsewhere with other people and are related to physical and social Presence.

Sheridan [27] used the term "Virtual Presence" to describe the feeling of "being present" caused by virtual reality technologies. When defining this term, Sheridan claimed to be possible to make a clear difference between virtual Presence (e.g. the sensation of Presence in a virtual environment), with the notion of telepresence, which is associated with teleoperation systems, as Minsky initially approached it. The delivery of Presence is closely tied to an understanding of consciousness and, in particular, of the interplay of implicit and explicit factors in the

construction of human behavior and their relation with virtual space. Presence is constructed by the brain and expresses the consistency between the world model the brain maintains and the cues it is exposed to.

Presence has long been a critical concept in teleoperation and virtual reality (VR) and has been defined as the "sense of being in a virtual environment" [28]. However, it is not clear how this "sense" is generated, and it is not uncommon to see it explained with the notion of "the suspension of disbelieve", coined in the early 19th Century by the poet and philosopher Coleridge. In recent literature, the notion of Presence results from the interplay of both central and peripheral factors and that it should be assessed through many convergent measures that include measures of the subjective, physiological and behavioral state of the user [28, 29]. Therefore, Presence, induced by virtual or physical sources of stimulation, is governed by several principles that underlie human experience, creativity and discovery.

The constructions of a meaningful relationship between agents and environmental stimuli in a virtual space and the exploration of their interactions can be anchored on the proposition that "interactive media within mixed reality environments induces an agent coupling with the space and it is defined as the sensing of Presence" [30]. Presence can also be studied as the relation between the implied identity of an organism/agent and its environment within the following perspectives: i) self-environmental (the agent exists in relation to the environment); ii) virtual objects (the object exists in relation to the agent); iii) social (the other agents exist in relationship to the agent). Physical Presence is when someone feels they are physically somewhere.

Presence also indicates that there are essential inputs for the construction of self-referral agencies. Thus, it is essential to deploy methodological efforts focusing on interactive media within a mixed reality environment to study the constructions of meaningful relationships between agents and virtual stimuli [31, 32]. The assumption is that the interaction of an agent or group of agents with an immersive space, using various interactive devices, indicates how these processes affect their behavior and the meaning that they construct. Such experience was approached by the interactive installation Ada: intelligent space developed and exposed at the Swiss National EXPO.02 [33].

Therefore, it is possible to project that the audience and musicians can feel present in the same space as the virtual robots. The challenge of devising a human-accessible mixed reality environment where humans and virtual robots are performing music together is that the interactions within such an environment dynamically shape musical performance. Thus the musician's virtual Presence is affected by the virtual robotic behavior. Both environments (virtual and physical) can access and influence each other, establishing an interaction. This possibility gives equal importance to both the physical and the virtual place in the performance outcome. In this exchange, the virtual world provides for limitless expression, and the real world defines physical grounding and the boundaries of interpretation. The virtual robots act in the virtual space, generate and react to sounds and human gestures. The musicians, in turn, generate meaning in the acoustic space and virtual one from these interactions with the virtual robots.

For our article's purpose, we will refer only to the term "Presence." In our case, we discuss the possibility of developing systems in which virtual robots interact with live musicians. We intended to study how such interaction affects the notion of real-time performance and how the inclusion of virtual robots can also enlarge musical aesthetic possibilities. In this way, virtual robots can be seen as a way to move from the notion of interactive performance to the concept of performance in a mixed reality in which musicians and virtual robots participate in a symbiotic process. We project that such experience would a) create a unified experience where virtual robots and musicians are merged in the virtual performance space (i.e. a mixed reality experience); b) the sound material generated by both evolve coherent in time; c) the resultant performance explore and exploit both implicit and explicit cues from musicians in their individual and collective interaction with the virtual robots; d) the use of novel multi-modal sensing and effector systems to boost interaction with and understanding of the dataflow generated during virtual robots-musicians interplay. In these processes, we also acquire that the virtual robots act as an adaptive sentient agent that helps humans explore creative spaces and discover novel patterns driven by both their implicit and explicit interactions.

Finally, it is important to notice here that the nature of musicians' interaction with digital technology involves behavior, perception, manipulation, and interaction. Perception leads musicians to identify and interpret acoustic and spatial relationships (e.g., among others) with the technological objects they are experiencing. For example, in live electroacoustic music, there is technological manipulation by the interpreters in other to sound processing. Otherwise, when someone changes the spatial location of an object in a virtual game, there is a level of experience in which the gamer perceives and acts on the virtual space. If users and virtual objects affect each other, the experience expands the physical world; therefore, the interaction occurs between these two domains, physical and virtual.

Thus, in our article's proposal, the musical interaction with virtual robots provides the exchange of musical information in real-time and induces responsive actions between robots and musicians. In this way, we project such musical experience to bring new meanings to performance experiences, such as a mixed music concert with the Presence of virtual robots.

As we discuss in this section, virtual robots can provide a sense of Presence, similarly to physical robots. This makes virtual robots a rich field for exploring interactions, which can foster – without physical parts – musical interactions similar to those provided by physical robots. Next, we discuss the current technological tools that can help achieving virtual robotic musicianship.

4. CURRENT TECHNOLOGY FOR VIRTUAL MUSICAL ROBOTS

The design of musical robots can be divided into three parts: a) establishing/developing the controlling interfaces, which ultimately supports the interaction between the musicians and the robots; b) designing the communication strategy, such as the protocols, redundancies, bandwidth, etc; and finally, c) designing and building the musical robot itself. Each of these aspects is discussed in a separate subsection, as follows.

4.1 Controlling robots

The interaction design for a particular robot is unique. It is virtually impossible to anticipate all the features a robot will have.

Interfaces to control musical robots can vary from custom-build software, wearable devices, MIDI-controllers, APIs, up to live programming languages - the latter with a very interesting application to real-time music performance. An important aspect to be considered is that programming languages specialized for sonic purposes, such as those that manipulate oscillators, filters, and other electronics for sound production, might not be the ideal candidate for the job, even though it usually provides OSC support. The reason behind this idea, is that musical robots do not manipulate sound signal directly ally. Usually, it does by manipulating the mechanics that produce the acoustic signal, meaning that controlling robots in more similar to conducting an orchestra than the usual electronic music live-coding. While the first activity uses gestures to communicate musical actions (musical terms) the latter use commands to manipulates the signal itself. In other words, robotic music performance has particularities that mimic human music performance. In that sense, the Octopus Music API [34], written in Java, is a viable option beacuse it models the performance using musically-related terms.

Octopus provides classes within three categories: a) musical data structures; b) musical data interpreters (performers); c) instrument classes. The musical pieces are structured using the classes of the musical data structures, similarly to a musical score. Such "object" is then played by the different levels of interpreters. A Musician is the Interpreter of the highest level, meaning that it "knows" how to play the piece in its crude form. A Guitarist is an example of an low-level Interpreter, meaning that it adapts the piece (in its persistent form) to their own capabilities, including restrictions of the its Instrument, in that case, the Guitar (instrument category). This metaphor is adequate to a Robotic performance, because, in this scenarion, a Robot can be seen as an instrument.

A major drawback of using an Java API such as Octopus is that the developed algorithm must be "compiled" before the program make any sound. In addition, Java is known to be a verbose language so, even if it possible to run in real-time using a REPL console (jshell), adjustments must be made to improve writability and make it more user-friendly to the final user (i.e. musicians) and this has been done using a musicality centred approach to the interaction design.

4.2 Communication protocols

The RoboMus framework [35] covers the whole musical robot production but has a special focus on its communication side. The RoboMus framework uses a predefined musical message format, built over the OSC protocol, to enable robot control and mutual collaboration.

It goes further by proposing macro and micro synchronization strategies by the use of a Musical Message Synchronization Server (MMSS) and onboard algorithm, respectively. Synchronization is particular sensitive matter considering that multiple robots are generally used in real-time music performance [14], and each robot takes a different time to react to its controller's inputs.

In order to support the interaction with all sort of musical robots, even those yet to be built, the RoboMus Framework used a handshake approach that allow the robots to introduce themselves to their human-performer by communicating all the "actions" they can perform using Open Sound Control (OSC), thus, any "client" that supports it could be used. This is as far as the RoboMus can go but it does not necessarily guarantee a good user experience in controlling the robots in musical activities. This, however, is out of the scope of this paper.

4.3 Robot design and physical simulation

Current physical modelling engines have proven powerful tools for real-time simulation and gaming. Tools specifically catered for robotics have added other functionalities to it, such as directly simulating specific types of engines, joints, and sensors. They have the advantage of corresponding to real-world objects, henceforth their behavior is predictable and understandable.

We give special attention to the Webots [36, 37], which is an open-source and multi-platform software. It provides a large asset library with pre-built robots, sensors, actuators, objects, and materials. Also, Webots allows programming the simulations in many languages: C, C++, Python, Java, MATLAB, or using a Robot Operating System API. Because it allows programming in several languages, it can use their libraries to send and receive information to and from the simulation using usual musical protocols, such as OSC or MIDI.

We used Webots to create a musical simulation to evaluate some of its possibilities. In this simulation, whose environment in shown in Figure 1, a generic robot moves within a limited space with objects. When it collides with an object, a sound is played. This robot musicality is similar to that found in Roboser [6].

We used a virtual sensor to measure the impact force, so that stronger collisions can generate stronger sounds. This measured impact force is sent to a SuperCollider script, which in turn synthesizes sounds using a physical model.

This short simulation raises an interesting dichotomy regarding the virtual robots: they are simulated within a physical modelling toolbox, which emulates realistic behavior; however, their output depends on musical composition and sound design, which can be (or not) realistic depending on the composer's intentions. Realism could

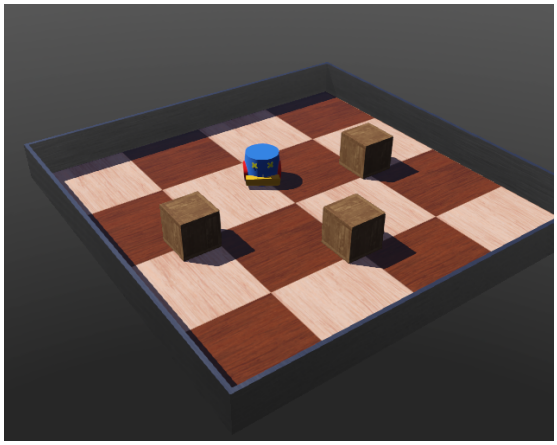


Figure 1. Robot simulation. The mobile robot (blue cylinder) navigates through the environment, and when it collides with an obstacle (brown boxes), the system emits a sound.

be even improved by using acoustic virtual reality techniques [38], which allow emulating object positioning in a sound scene that can be reproduced either via headphones or loudspeakers. In other words, this allows us to raise questions about bringing to the realistic physical model, and then the ability to enjoy artistic freedom, since the accessibility provided by virtuality cross barriers of reality, but allow verisimilitude even with non or ultra-realistic artistic options.

Next, we present concluding remarks.

5. CONCLUSION

In this position paper, we discuss the difficulties and possibilities brought by using virtual robots instead of physical ones in the context of musical performance. Nowadays, this change is possible because of the technologies related to physical simulation, and it can be greatly facilitated by previous work on tools for robotic musicianship.

Virtual robots can be effective as musical performance tools because they can foster Presence, that is, the sense that humans and virtual robots share the same environment. This happens because the feeling of Presence happens due to particular fluxes of consciousness, which do not necessarily depend on a physical presence. This phenomenon has been greatly explored in the last few years in the context of virtual reality, and it could be used to foster meaningful experiences in virtual robotic musicianship.

Also, nowadays, there are many tools and techniques that can help building virtual musical robots. These tools range from robot simulators to specific frameworks for music, and they can be explored even faster because the development of virtual robots is much faster and cheaper than that of physical ones.

Virtual robots are, also, more economically viable and environmentally-friendly than their physical counterparts. They do not require buying high-quality parts or finding ways to dispose of old ones. Moreover, they do not wear off with time, and they can be easily upgraded if needed.

Finally, because they exist as software, they allow collaborative work even during the social isolation required by the COVID19 pandemic. Furthermore, they allow larger, world-wide collaborations to happen, as it is the case of any piece of software. Furthermore, the technology stack used for the virtual robots can be revisited, and solutions different from those discussed in Section 4 can be used in a more integrated way than the one shown in this work.

For this reason, we propose that virtual robots can be effective tools for music making, and can be more viable than physical ones. Virtual robotic musicianship draws challenges of their own, such as questions regarding maintaining the environment's verisimilitude, designing the interactions and sounds synthesis related to the robot, or evaluating the sense of presence they could foster in musical interactions. All of these questions can be prolific fields for future explorations in art and research.

6. REFERENCES

[1] K. Tatar and P. Pasquier, "Musical agents: A typology and state of the art towards musical metacreation," *Journal of New Music Research*, vol. 48, no. 1, pp. 56–105, 2019. [Online]. Available: https://doi.org/10.1080/09298215.2018.1511736

[2] E. Singer, J. Feddersen, C. Redmon, and B. Bowen, "Lemur's musical robots," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, Hamamatsu, Japan, 2004, pp. 181–184. [Online]. Available: http://www.nime.org/proceedings/2004/nime2004_181.pdf

[3] G. Weinberg and S. Driscoll, "The design of a robotic marimba player: Introducing pitch into robotic musicianship," in *Proceedings of the 7th International Conference on New Interfaces for Musical Expression*, ser. NIME '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 228–233. [Online]. Available: https://doi.org/10.1145/1279740.1279786

[4] J. Solis, A. Takanishi, and K. Hashimoto, "Development of an Anthropomorphic Saxophone-Playing Robot," in *Brain, Body and Machine*, J. Angeles, B. Boulet, J. J. Clark, J. Kövecses, and K. Siddiqi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 175–186.

[5] L. Maes, G.-W. Raes, and T. Rogers, "The man and machine robot orchestra at logos," *COMPUTER MUSIC JOURNAL*, vol. 35, no. 4, pp. 28–48, 2011.

[6] J. Manzolli and P. F. M. J. Verschure, "Roboser: A real-world composition system," *Computer Music Journal*, vol. 29, no. 3, pp. 55–74, 2005. [Online]. Available: https://doi.org/10.1162/0148926054798133

[7] M. Bretan and G. Weinberg, "A survey of robotic musicianship," *Communications of the ACM*, vol. 59, no. 5, pp. 100–109, 2016.

[8] G. Weinberg and S. Driscoll, "Toward robotic musicianship," *Computer Music Journal*, vol. 30, no. 4, pp. 28–45, 2006. [Online]. Available: http://www.jstor.org/stable/4617982

[9] A. Kapur, E. Singer, M. S. Benning, G. Tzanetakis, and T. Trimpin, "Integrating HyperInstruments , Musical Robots & Machine Musicianship for North Indian Classical Music," in *Proceedings of the International Conference on New Interfaces for Musical Expression*. Zenodo, Jun. 2007, pp. 238–241. [Online]. Available: https://doi.org/10.5281/zenodo.1177137

[10] J. Long, D. Carnegie, and A. Kapur, "The closed-loop robotic glockenspiel: Improving musical robots with embedded musical information retrieval," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, vol. 16. Brisbane, Australia: Queensland Conservatorium Griffith University, 2016, pp. 2–7. [Online]. Available: http://www.nime.org/proceedings/2016/nime2016_paper0002.pdf

[11] M. Bretan and G. Weinberg, "A survey of robotic musicianship," *Commun. ACM*, vol. 59, no. 5, p. 100–109, Apr. 2016. [Online]. Available: https://doi.org/10.1145/2818994

[12] G. Hoffman and G. Weinberg, "Gesture-based human-robot jazz improvisation," in *2010 IEEE international conference on robotics and automation*. IEEE, 2010, pp. 582–587.

[13] R. V. Rooyen, A. Schloss, and G. Tzanetakis, "Voice coil actuators for percussion robotics," in *Proceedings of the International Conference on New Interfaces for Musical Expression*. Copenhagen, Denmark: Aalborg University Copenhagen, 2017, pp. 1–6. [Online]. Available: http://www.nime.org/proceedings/2017/nime2017_paper0001.pdf

[14] H. Camporez, J. Silva, L. Costalonga, and H. Rocha, "Robomus: Robotic musicians synchronization," in *Anais do X Workshop de Música Ubíqua (UbiMus 2020)*, 2020, p. 72.

[15] N. Yang, R. Savery, R. Sankaranarayanan, L. Zahray, and G. Weinberg, "Mechatronics-driven musical expressivity for robotic percussionists," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, R. Michon and F. Schroeder, Eds. Birmingham, UK: Birmingham City University, Jul. 2020, pp. 133–138. [Online]. Available: https://www.nime.org/proceedings/2020/nime2020_paper26.pdf

[16] R. A. Vieira and F. L. Schiavoni, "Fliperama: An affordable arduino based midi controller," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, R. Michon and F. Schroeder, Eds. Birmingham, UK: Birmingham City University, Jul. 2020, pp. 375–379. [Online]. Available: https://www.nime.org/proceedings/2020/nime2020_paper73.pdf

[17] R. Widmer, H. Oswald-Krapf, D. Sinha-Khetriwal, M. Schnellmann, and H. Böni, "Global perspectives on e-waste," *Environmental impact assessment review*, vol. 25, no. 5, pp. 436–458, 2005.

[18] R. Thomas, "Kurfess," 2005.

[19] T. Wang, S. Hu, J. Xu, D. Yan, and J. Bi, "Simulation Design and Application of Music Playing Robot Based on SolidWorks," in *2009 International Conference on Measuring Technology and Mechatronics Automation*, vol. 2, Apr. 2009, pp. 339–342, iSSN: 2157-1481.

[20] M. Shahab, A. Taheri, S. R. Hosseini, M. Mokhtari, A. Meghdari, M. Alemi, H. Pouretemad, A. Shariati, and A. G. Pour, "Social Virtual Reality Robot (V2R): A Novel Concept for Education and Rehabilitation of Children with Autism," in *2017 5th RSI International Conference on Robotics and Mechatronics (ICRoM)*, Oct. 2017, pp. 82–87, iSSN: 2572-6889.

[21] P. Chevalier, V. Vasco, C. Willemse, D. De Tommaso, V. Tikhanoff, U. Pattacini, and A. Wykowska, "Upper limb exercise with Physical and Virtual Robots: Visual Sensitivity Affects Task Performance," *Paladyn, Journal of Behavioral Robotics*, Jan. 2021. [Online]. Available: https://doi.org/10.1515/pjbr-2021-0014

[22] H. A. Camporez, T. S. Mota, L. Costalonga, and H. Rocha, "Protótipo robótico de baixo custo para performance musical em violão acústico."

[23] L. L. Costalonga, E. de Aguiar, D. Coura, and M. Neves, "Campus vivo–instalações artísticas e artefatos culturais computadorizados," *ScientiaTec*, vol. 2, no. 2, pp. 75–84, 2015.

[24] M. Lee and K. Presence, "Explicated," *Communication Theory*, pp. 27–50, 2004.

[25] M. Slater and M. Usoh, "Representations systems, perceptual position, and presence in immersive virtual environments," *Presence: Teleoperators and Virtual Environments*, vol. 2, p. 221–233, 1993.

[26] M. Minsky, "Telepresence," *Omni*, vol. 2, pp. 45–51, 1980.

[27] T. B. Sheridan, "Musings on telepresence and virtual presence," *Presence: Teleoperators and Virtual Environments*, vol. 1, p. 120–126, 1992.

[28] E. a. Dubois, "The engineering of mixed reality systems, springer verlag, london, isbn: 978-1-84882-723-5," vol. 2010.

[29] R. P. de Oliveira, D. C. P. de Oliveira, and T. F. Tavares, "Measurement methods for phenomena associated with immersion, engagement, flow, and presence in digital games," in *Simpósio Brasileiro de Jogos e Entretenimento Digital*, 2016.

[30] M. V. Sanchez-Vives and M. Slater, "From presence to consciousness through virtual reality," *Nat Rev. Neuroscience*, vol. 6, no. 4, pp. 332–9, 2005.

[31] U. e. a. Bernardet, "The experience induction machine: A new paradigm for mixed-reality interaction design and psychological experimentation," in *The Engineering of Mixed Reality Systems*. Springer, 2010, pp. 357–379.

[32] M. Inderbitzin, "S. et. al," *Social cooperation and competition in the mixed reality space eXperience Induction Machine (XIM). Virtual Reality*, vol. 13, p. 153–158, 2009.

[33] K. Wasserman, J. Manzolli, K. Eng, and P. F. M. J. Verschure, "Live soundscape composition based on synthetic emotions: Using music to communicate between an interactive exhibition and its visitors," *IEEE MultiMedia*, vol. 10, p. 82–90, 2003.

[34] L. Costalonga, "Biomechanical modelling of musical performance: Acase study of the guitar," Ph.D. dissertation, University of Plymouth, 11 2009.

[35] H. Camporez, T. Mota, E. Astorga, M. Neves, H. Rocha, and L. Costalonga, "Robomus: Uma plataforma para performances musicais robóticas," *Applications in Ubiquitous Music. São Paulo, SP: Editora ANPPOM*, 2018.

[36] Webots, "http://www.cyberbotics.com," open-source Mobile Robot Simulation Software. [Online]. Available: http://www.cyberbotics.com

[37] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotics Systems*, vol. 1, no. 1, pp. 39–42, 2004. [Online]. Available: http://www.ars-journal.com/International-Journal-of-Advanced-Robotic-Systems/Volume-1/39-42.pdf

[38] M. Vorländer, *Auralization: Fundamentals of Acoustics, Modelling, Simulation, Algorithms and Acoustic Virtual Reality (RWTHedition)*. Springer, 2007.

Proceedings of the 18th Sound and Music Computing Conference

9 788894 541540