

A System for Expressive Spectro-Spatial Sound Synthesis



Henrik von Coler

A System for Expressive Spectro-Spatial Sound Synthesis

vorgelegt von
M. Sc.
Henrik von Coler

An der Fakultät I - Geisteswissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften
Dr. rer. nat.
genehmigte Dissertation

Promotionsausschuss:

Vorsitzende: Prof. Dr. Birgit Beck (TU Berlin)
Gutachter: Prof. Dr. Stefan Weinzierl (TU Berlin)
Gutachter: Prof. Dr. Miller Puckette (University of California, San Diego)
Gutachter: Dr. Diemo Schwarz (IRCAM, Paris)

Tag der wissenschaftlichen Aussprache: 10. Dezember 2020

Berlin 2021

Abstract

The GLOOO system is a holistic digital musical instrument for spatial sound synthesis, considering a control device, a real time synthesis algorithm with a related modeling procedure and the integration of sound reproduction systems.

Statistical Spectral Modeling is introduced as a novel method for transferring a sample library of musical instrument sounds into a data model for the application in expressive synthesis systems. A custom sample library of anechoic violin sounds has been produced to generate the model used within this project.

Centerpiece of the instrument is a multi-channel synthesis software, implementing the *Statistical Spectral Synthesis* algorithm in C++. All spectral components of the synthesis software are dynamically routed to multiple outputs and sent to a third party spatial rendering software. This allows the realization of *Spectro-Spatial Sound Synthesis* - the integrated spectral synthesis of sound with dynamic spatial distribution properties, controlled through a set of shared control parameters.

Expressive control over synthesis and spatialization is enabled through a custom-designed haptic interface, developed in an interdisciplinary team. Within a user study, participants had the opportunity to create individual mappings between interface and spatial synthesis, using a visual programming environment. Control and synthesis parameters of the system as well as the mapping strategies of different users were investigated. Results show a preference for specific sensor units of the control device and for specific connections between control and synthesis parameters. Case examples reveal different mapping types, including the *high confidence mapper*, the *exploring mapper* and the *confused mapper*. The full system can be considered an environment for further studies on human-computer interaction in musical contexts.

Zusammenfassung

Das GLOOO System ist ein holistisches digitales Musikinstrument für räumliche Klangsynthese, in welchem ein Kontrollgerät, ein Synthesalgorithmus mit zugehörigem Modellierungsverfahren sowie die Einbindung von Lautsprechersystemen berücksichtigt werden.

Statistical Spectral Modeling wird als neue Methode vorgestellt, um eine Sample-Bibliothek von Musikinstrumentenklängen in ein spektrales Klangmodell für die Verwendung in expressiven Echtzeit-Synthesesystemen zu überführen. Eine Datenbank mit hallfreien Violinenklängen, welche explizit auf dieses Verfahren zugeschnitten ist, wurde für die Nutzung mit dem Analysesystem produziert.

Zentraler Bestandteil des Instruments ist eine Synthese-Software, in welcher der *Statistical Spectral Synthesis* Algorithmus in der Programmiersprache C++ implementiert ist. Diese Software bietet eine hohe Anzahl von Ausgangskanälen für die einzelnen Komponenten der spektralen Synthese. Die Ausgänge des Synthesizers werden an eine vorhandene Rendering-Software geleitet, mittels der sich die spektralen Komponenten dynamisch verräumlichen lassen. Der Begriff *Spectro-Spatial Sound Synthesis* beschreibt die dadurch ermöglichte Form der Synthese, in der spektrale und räumliche Eigenschaften auf einer frühen Ebene durch einen gemeinsamen Strom von Kontrolldaten gesteuert werden.

Für die expressive Kontrolle von Synthese und Verräumlichung wurde von einem interdisziplinären Team ein haptisches Interface entwickelt. Im Rahmen einer Nutzerstudie wurden von Teilnehmer*innen über eine visuelle Programmierungsumgebung individuelle Verknüpfungen zwischen dem Interface und dem räumlichen Synthesesystem hergestellt. In der Auswertung der Ergebnisse zeigen sich Präferenzen für bestimmte Verbindungen und Komponenten des Interfaces. Darüber hinaus lassen sich die Strategien der Teilnehmer*innen während dieses Vorgangs analysieren. In Fallbetrachtungen wurden die Nutzertypen *High Confidence Mapper*, *Exploring Mapper* und *Confused Mapper* identifiziert. Das gesamte System eignet sich darüber hinaus für weitere Studien zur Mensch-Computer-Interaktion im musikalischen Kontext.

Related Publications

Conference Papers

The following peer reviewed conference contributions are related to the project presented in this dissertation.

von Coler, H., Lepa, S., & Weinzierl, S. (2020). User-defined mappings for spatial sound synthesis. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Birmingham, UK (Online)

von Coler, H. (2019b). Statistical sinusoidal modeling for expressive sound synthesis. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Birmingham, UK

von Coler, H. (2019a). A JACK-based application for spectro-spatial additive synthesis. In *Proceedings of the 17th Linux Audio Conference (LAC-19)*. Stanford University, CA, USA

von Coler, H., Götz, M., & Lepa, S. (2018). Parametric synthesis of glissando note transitions – a user study in a real-time application. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Aveiro, Portugal

von Coler, H. (2018). TU-Note Violin Sample Library – a database of violin sounds with segmentation ground truth. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Aveiro, Portugal

von Coler, H., Treindl, G., Egermann, H., & Weinzierl, S. (2017). Development and evaluation of an interface with four-finger pitch selection. In *Proceedings of the 142nd Audio Engineering Society Convention*. Berlin, Germany

von Coler, H. & Lerch, A. (2014). CMMSD: A data set for note-level segmentation of monophonic music. In *Proceedings of the AES 53rd International Conference on Semantic Audio*. London, UK

von Coler, H. (2014). Musical expressivity - considerations between sound synthesis, musicology and other disciplines. In *Proceedings of the 9th Conference on Interdisciplinary Musicology - CIM14* (pp. 122–125). Berlin, Germany

von Coler, H. & Röbel, A. (2011). Vibrato detection using cross correlation between temporal energy and fundamental frequency. In *Proceedings of the 131st Audio Engineering Society Convention*. New York, USA

Master's Theses

The following master's theses were supervised within the scope of this work and contributed to the development of the project:

Treindl, G. (2016). *Entwicklung und Evaluation eines Controllers zur Tonhöhensteuerung über Vier-Finger-Kombinationen* (Master's thesis, TU Berlin)

Wiemann, B. (2017). *Implementation of an additive sound synthesis for an electronic musical instrument* (Master's thesis, TU Berlin)

Götz, P. M. (2018). *Analysis and synthesis of control parameters in note transitions* (Master's thesis, TU Berlin)

Schmied, A. (2018). *Development and stress testing of the BinBong MKII* (Master's thesis, TU Berlin, Berlin, Germany)

Contents

I Introduction & Related Work

1 Project Outline	14
1.1 Project Introduction	14
1.2 Structure of this Thesis	16
2 Expressivity	17
2.1 A Working Definition	17
2.2 Expressivity and the Violin	19
2.3 Articulation	20
2.3.1 Temporal Measures	20
2.3.2 Bowing Patterns	22
2.3.3 Glissando	23
2.4 Vibrato	24
2.4.1 Different Types of Vibrato	24
2.4.2 Spectral Envelope Modulations	25
2.4.3 Vibrato Analysis	26
3 Sound Synthesis	29
3.1 A Taxonomy	29
3.2 Processed Recording	32
3.2.1 Origins	32
3.2.2 Granular Synthesis	32

3.2.3	<i>Concatenative</i>	33
3.3	Spectral Modeling	34
3.3.1	<i>Additive Synthesis</i>	34
3.3.2	<i>Evolution of Spectral Modeling</i>	34
3.3.3	<i>Deterministic Part</i>	35
3.3.4	<i>Residual Part</i>	36
3.3.5	<i>Window Size and Hop Size</i>	38
3.3.6	<i>IFFT Synthesis</i>	39
3.4	Expressive Synthesis Approaches	42
3.4.1	<i>Reconstructive Phrase Modeling</i>	42
3.4.2	<i>Spectral Concatenative</i>	43
3.4.3	<i>Excitation plus Resonances</i>	44
3.4.4	<i>Spectral Interpolation Synthesis</i>	44
3.4.5	<i>Removing the Time Axis</i>	45
3.4.6	<i>Extended Source-Filter Model</i>	45
3.4.7	<i>Statistical Parametric Speech Synthesis</i>	46
3.5	Spatial Sound Synthesis	47
3.5.1	<i>Spatialization & Diffusion</i>	47
3.5.2	<i>Spectral Spatialization</i>	48
3.5.3	<i>Spatial Sound Synthesis</i>	49
3.5.4	<i>Spectro-Spatial Sound Synthesis</i>	53
4	Control & Mapping	55
4.1	Interfaces in Digital Musical Instruments	55
4.2	Expressive Control of Sound Synthesis	58
4.2.1	<i>Historic Concepts for Expressive Control</i>	60
4.2.2	<i>Control in Classic Synthesizers</i>	60
4.2.3	<i>Alternative Control Principles</i>	61
4.2.4	<i>Partial Performance Synthesis</i>	64
4.3	Control and Spatialization	66
4.3.1	<i>Historical Viewpoints</i>	66
4.3.2	<i>Parameters of Spatial Control</i>	67
4.3.3	<i>Devices for Spatial Control</i>	69
4.4	Mapping in Digital Musical Instruments	73
4.4.1	<i>The DMI Model</i>	73
4.4.2	<i>Fusion-Mapping-Fission Model</i>	73

4.4.3	<i>Fingering Systems</i>	75
4.4.4	<i>User-Defined Mapping</i>	76

II The GLOO System

5	Components of the GLOO System	82
6	Sample Library	84
6.1	Libraries for Production and Research	84
6.2	Audio Content	86
6.2.1	<i>Single Sounds</i>	86
6.2.2	<i>Two-Note Sequences</i>	87
6.2.3	<i>Solo</i>	89
6.3	Recording Setup and Procedure	91
6.3.1	<i>Environment</i>	91
6.3.2	<i>Instructions</i>	93
6.4	Segmentation	95
6.4.1	<i>The Note-Transition-Rest Paradigm</i>	95
6.4.2	<i>Tools and Procedure</i>	96
6.4.3	<i>Label Data</i>	98
7	Analysis and Modeling	99
7.1	Overview	99
7.2	Spectral Modeling	101
7.2.1	<i>Pitch Tracking</i>	101
7.2.2	<i>Partial Tracking</i>	101
7.2.3	<i>Residual Modeling</i>	106
7.3	Statistical Modeling	108
7.3.1	<i>Trajectory Decomposition</i>	109
7.3.2	<i>Stateless Statistical Modeling</i>	114
7.3.3	<i>Markovian Statistical Modeling</i>	118
7.4	Transition Modeling	124
7.4.1	<i>Attack</i>	124
7.4.2	<i>Release</i>	124

7.4.3	<i>Glissando</i>	128
7.5	Final Data Set	130
7.5.1	<i>Deterministic Data</i>	130
7.5.2	<i>Statistical Data</i>	130
8	Synthesis System	133
8.1	Statistical Spectral Synthesis	133
8.1.1	<i>Timbre-Space Interpolation</i>	133
8.1.2	<i>Mean Mode</i>	135
8.1.3	<i>Stateless Inverse Transform Sampling Synthesis</i>	135
8.1.4	<i>Markovian Inverse Transform Sampling Synthesis</i>	137
8.2	Spectral Dispersion	139
8.2.1	<i>Uniform Dispersion</i>	139
8.2.2	<i>Discrete Dispersion</i>	140
8.2.3	<i>Filter Bank Dispersion</i>	140
8.3	Implementation	143
8.3.1	<i>Dependencies and Runtime Environment</i>	143
8.3.2	<i>GLOOO Classes and Structure</i>	144
8.3.3	<i>Voice Management</i>	148
8.3.4	<i>Synthesis Process</i>	149
8.3.5	<i>GLOOO Configuration</i>	152
8.3.6	<i>Synthesis Parameters</i>	154
8.4	Measurements	156
8.4.1	<i>Pitch Sweeps</i>	156
8.4.2	<i>Intensity Sweeps</i>	161
9	Control & Mapping	164
9.1	BINBONG MK I	164
9.1.1	<i>Motivation and First Experiments</i>	164
9.1.2	<i>Design Studies</i>	166
9.1.3	<i>First Prototype</i>	167
9.2	BINBONG MK II	169
9.2.1	<i>Improvements to the First Version</i>	169
9.2.2	<i>Housing</i>	170
9.2.3	<i>Electronics</i>	171
9.2.4	<i>FSR Units</i>	171

CONTENTS

9.2.5	Valves	172
9.2.6	Pad	174
9.2.7	Ribbon	175
9.2.8	IMU Parameters	176
9.2.9	Output Values	178
9.3	Receiver & Mapping Software	180
9.3.1	Mapping in Spatial Sound Synthesis	180
9.3.2	BINBONG Receiver	180
9.3.3	Spatial Distribution	181
9.3.4	Synthesis Control	183
9.3.5	Mapping Environment	184

III User Study & Conclusion

10	User Study	186
10.1	Background	186
10.2	Test Setup	187
10.2.1	Space and Equipment	187
10.2.2	The Runtime System	187
10.3	Method	190
10.3.1	Mapping Stage	190
10.3.2	Gesture Stage	192
10.3.3	Additional Surveys	194
10.4	Results	195
10.4.1	Sample	195
10.4.2	GMSI	196
10.4.3	UEQ	197
10.4.4	Gesture Survey	198
10.4.5	Mapping Process	200
10.4.6	Individual Connections to Rendering Parameters	201
10.4.7	Individual Connections to Control Parameters	203
10.4.8	Final Mappings	205
10.5	Discussion	208
10.5.1	Sample and Surveys	208

10.5.2	Text Response Examples	208
10.5.3	Final Mappings	209
10.5.4	Mapping Process	209
10.5.5	Mapping Strategies	210
11	Conclusion	213
11.1	The Modeling Approach	213
11.2	Synthesis Software	214
11.3	Spatial Synthesis	215
11.4	Control & Mapping	215
11.5	User-Defined Mappings	216
<hr/>		
	References	217
	Index	245
	List of Figures	252
	List of Tables	254

Appendix

I	Single Sounds List	255
II	Two-Note List	263
III	Configuration File	271

Part I

Introduction & Related Work

Project Outline

1.1 Project Introduction

The project presented in this thesis started out with a scope on signal processing and methods for sound analysis and synthesis, with the goal of making sample-based synthesis approaches more expressive. Yet, it grew into a complete system for interactive sound synthesis, respectively a musical instrument. Many different aspects are thus part of the project and will be treated in this document, each with the necessary depth to provide an understanding of the instrument and its idiosyncrasies. Figure 1.1 illustrates how the relevant components of the project are related.

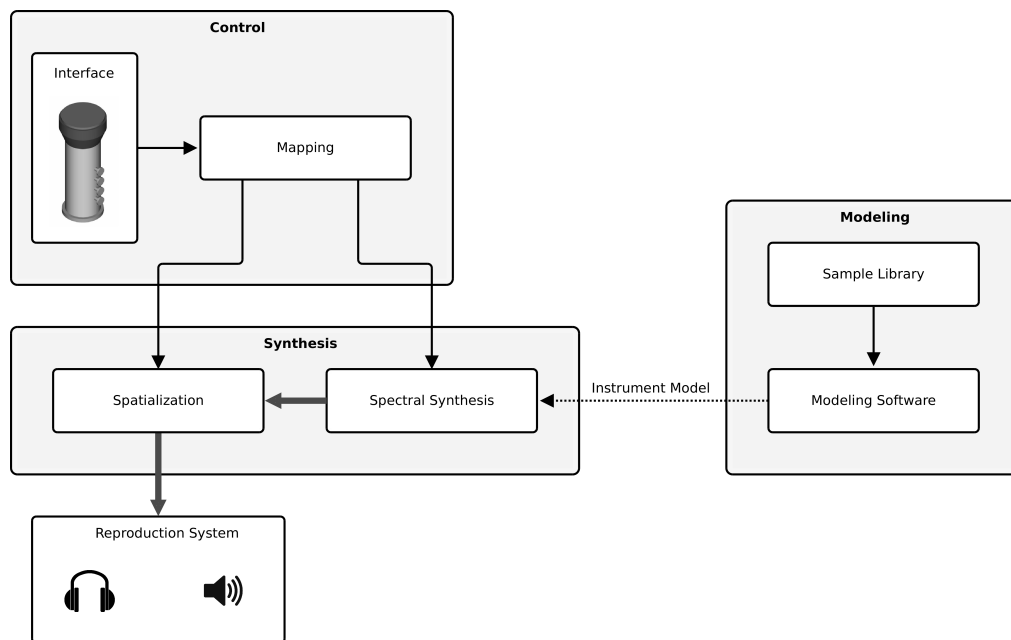


Figure 1.1: Relevant components of the project and their relation.

The proposed approach for sound synthesis can be briefly described as a combination of sample-based synthesis and an extended spectral modeling technique. It is thus inherently connected to an analysis algorithm. First practical steps focused on the acquisition and preparation of sound material for the modeling system. A custom sample library was created for this purpose, covering specific aspects not found in existing data sets. Based on established methods for spectral modeling, a novel statistical approach has been developed, with the aim of increasing the expressive capabilities. Using a statistical layer for modeling the temporal behavior of partial and noise trajectories, basic timbral qualities can be learned from previously recorded samples. Sound analysis and modeling is programmed in Matlab scripts, which provide the instrument model for the synthesis.

Centerpiece of the instrument and the development efforts is a real-time synthesis system, implementing the *Statistical Spectral Synthesis* algorithm in C++. The resulting software runs as a JACK client and can hence be integrated into professional Linux audio systems for research and performance applications. It is entirely controlled via the OSC protocol to interact with other processes.

The synthesis software offers multiple audio outputs for individual spectral components. These outputs are routed to an existing spatialization software to individually distribute the spectral components of the sound to loudspeaker systems or headphones. This procedure is referred to as *spectro-spatial sound synthesis*.

Electronic instruments for expressive musical performance rely on control interfaces, granting access to the sound generation and manipulation. A haptic interface has thus been co-developed, considering the requirements of the synthesis system. The wireless device allows the joint control of spectral and spatial parameters through multiple force-sensitive and orientation sensors. A mapping stage connects the sensor data of the interface to both the spectral synthesis and the spatialization software.

The above introduced components constitute a musical instrument, which is of course intended to be used in performance situations. However, its primary focus is the use as both a research tool and an object of research. Especially the connection between the control interface, the synthesis algorithm and the spatialization is an open question. This problem, also known as *mapping*, has a large influence on the expressive capabilities of the instrument and on the resulting user experience. The developed instrument can help to explore this aspect, due to its flexibility.

1.2 Structure of this Thesis

The remainder of this introductory Part I introduces the core problems and gives an overview of related work in the relevant fields. This includes considerations from musicology and music psychology for an investigation of the concept of *expressivity* and its meaning for the goals of this thesis in Chapter 2. The topic of sound synthesis is introduced with a focus on expressive spectral modeling and spatial sound synthesis in Chapter 3. In Chapter 4, the control of digital musical instruments is treated, emphasizing interfaces for synthesizers and spatilization systems, as well as mapping as an integral part of digital musical instruments.

The following Part II presents the individual components of the instrument, developed in the course of this project. The sample library is presented in Chapter 6, with focus on the content and recording techniques, followed by Chapter 7 with details on signal analysis and the modeling algorithm. This also includes the detailed description of the novel *statistical spectral modeling* approach. Chapter 8 explains the synthesis algorithm, as well as the implementation of the real-time software and the operating principle of *spectro-spatial synthesis*. Chapter 9 reports on the development of the dedicated haptic interface, including the full history and the recent prototype.

The final Part III includes the user study for exploring user-defined mappings in Section 10. Summary, conclusion and an outlook are presented in Chapter 11.

Chapter 2

Expressivity

This chapter introduces the concept of expressivity in relation to this project, starting with a working definition in Section 2.1. Specific aspects of musical expressivity and expressive synthesis of the violin are discussed in Section 2.2. Articulation and vibrato are treated as expressive components of specific interest in Section 2.3, respectively Section 2.4.

2.1 A Working Definition

A central aspect of this thesis is the implementation of a system capable of expressive sound synthesis. However, in the fields of engineering, musicology and electroacoustic music, the term *expressivity*, sometimes referred to *expressiveness*, is used in different ways. It is thus beneficial to establish a working definition for the specific problems of this project, based on viewpoints and definitions found in relevant disciplines. Most of these considerations are concerned with the parameters of melody and rhythm in classical Western music. The expressive dimensions of spatial performance are not covered.

Among the most prominent representatives in the history of research on musical expressivity is Carl E. Seashore. He describes expressivity as the “artistic expression of feeling” in music and, more technically, as “the deviation from the rigid, the exact, the regular” (Seashore, 1938, p. 9). According to Seashore, musical expression is produced through changes in pitch, intensity, duration and timbral qualities.

Within his *Generative Theory of Tonal Music*, Lerdahl defines “essentially quantized pitches and rhythms with dynamic and timbral attributes” (Lerdahl, 2009, p.2) as the *musical surface*. This approach is congruent to Seashore’s model, leaving creative means beyond this quantized grid as what is referred to as *expressive musical content* in the remainder of this thesis.

According to London (2000) the expressivity of musical performance is on the one hand linked to specific capabilities of a performer to play with a certain amount of virtuosity, on the other

hand it refers to the ability to embed emotions in a performance. London further distinguishes between the expressive properties of sound, defined by measurable quantities like loudness or roughness and expressive properties of music, including musical syntax and context. In engineering, *expressiveness* is also understood as the ability or capacity of a sound synthesis system to convey emotions. According to Arfib, Couturier, and Kessous (2005), expressive digital musical instruments must simply be able to produce expressive sounds, regardless of the driving gesture or intention.

Another definition of musical expressivity frequently found in the discourse understands it as the deviation of the performance from the score (Palmer & Hutchins, 2006; De Poli, 2004). Expressive musical performance is understood as “the deliberate shaping of the music by the performer, the imposing of expressive qualities onto an otherwise ‘dead’ musical score via controlled variation of parameters such as intensity, tempo, timing, articulation” (Goebel, Dixon, De Poli, Friberg, & Bresin, 2008, p.179). In the scope of this thesis this definition is rather misleading, since it seems more suitable for describing the concept of interpretation versus notation. After all, a composer can be as specific as desired and include very precise instructions on expressive features in scores. In his work *YPSILON* (Stockhausen, 1989), for example, Karlheinz Stockhausen gives detailed instructions on the use of amplitude vibrato, glissando and draws pitch contours beyond traditional notation. This can be referred to as *composed expressive musical content*.

According to Gabrielsson and Juslin (1996), the mechanisms of expressing emotion through basic compositional means, respectively the musical grid, such as mode and tempo, have been investigated extensively and understood to a certain degree. A large body of work has used synthesized musical performances to investigate the influence of musical cues like *mode*, *tempo*, *pitch level*, *rhythm quality*, *loudness*, *pitch variation* and others on the rating of expressed emotions (Eerola, Friberg, & Bresin, 2013). Within an experiment, Gabrielsson and Juslin showed that the intentions of musicians to express different emotional states result in significant differences in tempo, dynamics and spectrum. The relations between expressive musical content and expressed emotion, however, are less explored, supposedly because these are harder to quantify.

Moving on from the concepts of Seashore, which basically rely on measured signal attributes, Juslin (2003) regards expressivity as a composition of perceptual qualities that are comprised of the acoustical properties of the music and the subjective impressions of the listener. This expands the focus and Juslin lists the following factors which have influence on the expression in a musical performance: *piece-related*, *instrument-related*, *performer-related*, *listener-related* and *context-related*.

Piece-related attributes refer to what has before been declared as musical grid. The *listener-related factors* like the listener’s current mood, music expertise, musical preferences and others are not relevant for the scope of this thesis, as are the *context-related* factors, including acoustics, sound technology, listening context and larger cultural and historic setting. However, in terms of musical instruments they contribute to the transfer of expressive musical content.

A factor of interest in the scope of this thesis is the *instrument-related* one, including the acoustic parameters, the instrument-specific aspects of timbre and pitch, as well as the technical difficulties of the instrument. The second factor of interest is *performer-related*, including the performer’s technical skill, the performer’s motor precision and the performer’s emotional state

while playing. Combining these two factors is the central aspect when designing novel musical instruments.

Juslin proposes the GERMS model of musical expressivity. The model is assembled of five components, namely Generative Rules (G), Emotional Expression (E), Random Variability (R), Motion Principles (M) and Stylistic Unexpectedness (S). Generative rules are used by performers for emphasizing the underlying musical structures and make them clear for the audience. Emotional expression is applied by musicians to render a performance in a specific emotional way. Random variability refers to that part of musical performance which is caused by the shortcomings of our motor-system, resulting in random fluctuations and a slight unpredictability. Motion principles are based on the assumption that music and motion are closely related and that change in music induces an experience of motion. Stylistic unexpectedness includes the deviation from stylistic expectations, related to established performance conventions. Thus, psychological tension can be created by violating expectations. In the development of expressive sound synthesis systems, the random variability is one of the most important aspects. It combines a fully controllable instrument with the stochastic elements for making it sound both expressive and natural.

Not only music which is actually performed by a musician on stage or in a recording studio is able to contain expressive musical content. In the field of computer music, Oppenheim and Wright (1996) introduce the term *composed expressions*, for expressive musical content which is generated when creating fixed media compositions in the studio. Although in this stream of thought, the process of composing in the studio is regarded an act of performance, it is not necessarily linked to a classical performance situation. In this case, expressivity becomes a phenomenon communicated only through the sonic qualities of the resulting piece.

Regarding the scope of this work, the definition of expressivity as the *deviation of the rigid* is well suited. From the performer's viewpoint, a system for expressive sound synthesis must be able to render streams of control parameters into sound in a meaningful way. It must react to expressive gestures with expressive sound production. From the instrument's point of view, expressive musical synthesis must also render the inherent deviations from a pure and rigid sound on a microstructural level. This is relevant for the related expressive sound synthesis approaches, introduced in Section 3.4, but also for the interfaces for expressive control, discussed in Section 4.2. Their key principle is to allow musicians the application of these deviations through gestural control and signal processing.

2.2 Expressivity and the Violin

The analysis-synthesis system presented in this thesis was developed using a library of violin recordings. For several reasons, the violin is a popular instrument for the research on analysis-synthesis systems and expressive synthesis. It belongs to the excitation-continuous instruments, allowing the generation of long, sustained tones through continuing bow movement. This enables the application of various techniques during note transitions and throughout the sustain portion of the tone, increasing the expressive means of the instrument. Dannenberg and Derenyi

(1998) emphasize this relation for wind instruments with continuous excitation, where consecutive sounds are usually not independent. Excitation-continuous instruments represent a case of special interest in terms of expressive synthesis and many projects in sound analysis-synthesis thus focus on the violin (Hahn, 2015), violins and bassoons (Wager, Chen, Kim, & Raphael, 2017) or the singing voice (Bonada & Serra, 2007).

As a fretless chordophone, the violin has a continuous frequency scale for each string. This makes techniques like frequency vibrato and glissando important means of expression on this instrument. Microtonal deviations and fluctuations from exact pitches are essential features of a violin's sound.

The violin timbre is basically controlled through the three bowing parameters *pressure*, *speed* and *location* (Galamian, 1999, p. 73). Their combination leads to different styles of tone production. Galamian (1999) reduces the problem to two basic types. One is played with little pressure and high speed, located further away from the bridge, which produces a tone with a *light and loose character*. The second is played with more pressure and slower motion, at a location closer to the bridge. This results in a tone with a *quality of denseness and concentration*. The bowing parameters and the following timbral qualities are not considered by the synthesis system presented in this thesis.

2.3 Articulation

Just as in speech, articulation in music is a means for emphasizing and grouping single events, in order to support and produce structure. Articulation refers to the shaping of individual notes, in the remainder referred to as *intra-note articulations*, as well as to the shaping of note transitions and phrases, further referred to as *inter-note articulations*. In most cases, as for example in bowing patterns for string instruments, techniques affect both inter- and intra-note articulation. Articulation styles do not only constitute a significant part of the expressive repertoire of a performance, but they also account for the timbral qualities and the recognizability of instruments.

2.3.1 Temporal Measures

Inter-note articulations include the expressive musical content related to the shaping of transitions between consecutive notes. With the increasing possibilities of computer-aided means for the analysis of musical performances, detailed observations were possible. Strawn (1985, 1986) investigated transitions of orchestral instruments based on time varying spectra and power trajectories. The most general approach for classifying note transitions is the quantification of note durations, transition durations and the ratio between these intervals. Strawn introduces three basic modes for simplified transition models, shown in Figure 2.1. They include a mode where the notes do not show any interaction with a clear gap, one where the end of the first note's decay and the start of the following attack coincide, both referred to as *detached* in the remainder, and one with an overlap of decay and attack, referred to as *legato*. The time between the end of the first note's decay and the start of the following attack is referred to as *gap time*.

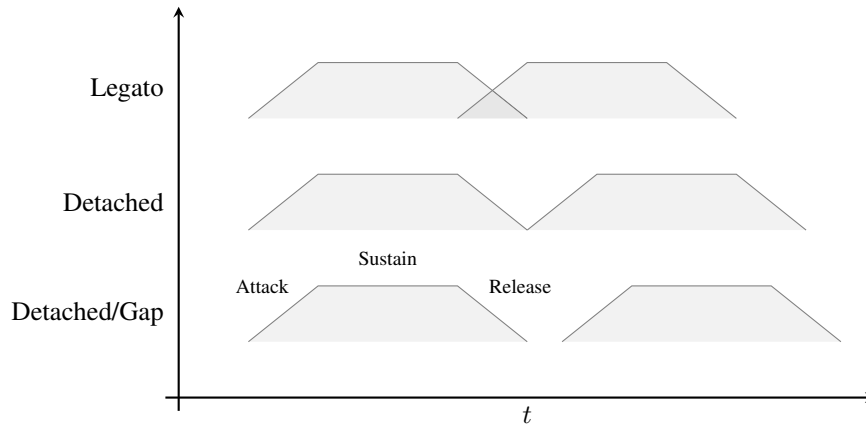


Figure 2.1: Three modes of inter-note articulation after Strawn (1985).

Based on this coarse classification, various measures have been proposed for quantifying parameters of musical transitions. Using different naming schemes, they usually take into account the durations of subsequent notes and the transition between them, as well as the contour of the energy and fundamental frequency trajectory within the transition segment.

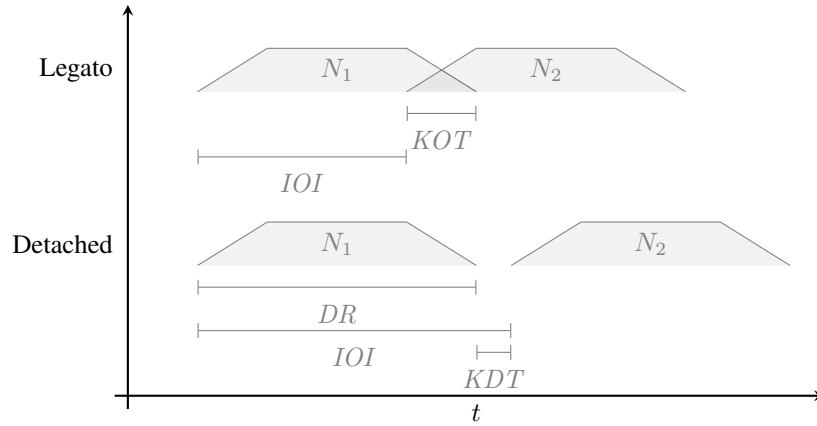


Figure 2.2: Durational parameters of notes and transitions after Bresin (2001).

Figure 2.2 shows relevant intervals and durations needed for obtaining several measures on musical transition. The established *Inter Onset Interval* (IOI) measures the time passed between two consecutive note onsets (Bresin, 2001; Friberg, Schoonderwaldt, & Juslin, 2007), not including any note or gap durations. Based on duration (DR) and IOI, Bresin (2001) introduces the *Key Overlap Time* (KOT) for articulation indices greater one (overlapping notes) and the *Key Detached Time* (KDT) for articulation indices below one, as included in Figure 2.2. The *Articulation Factor* (ART), also entitled *Legato Descriptor* (Maestre, Ramirez, Kersten, & Serra, 2009) or *Articulation Index* (Loureiro, Yehia, de Paula, Campolina, & Mota, 2009), is calculated as the ratio between note duration and inter onset interval:

$$ART = \frac{DR}{IOI} \quad (2.1)$$

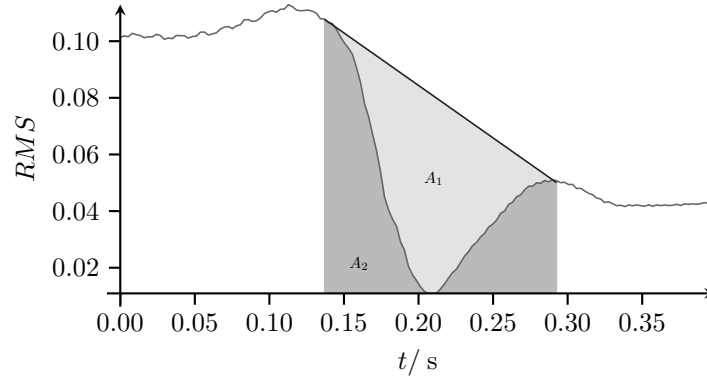


Figure 2.3: RMS trajectory of two-note item 123 from the sample library with areas for calculating the legato index.

The *Legato Index* (*LEG*) measures the energy drop during note transitions (Maestre & Gomez, 2005; Loureiro et al., 2009; Maestre et al., 2009). Figure 2.3 shows the energy trajectory of a two-note item from the sample library in the transition segment. Drawing a straight line between note offset and note onset in the energy trajectory results in two areas A_1 and A_2 . The legato index can be calculated as the ratio between A_1 and the sum of these two areas:

$$LEG = \frac{A_1}{A_1 + A_2} \quad (2.2)$$

Some phenomena of inter-note articulation occur at the boundaries of the enclosing notes. *Over-shoots* and *preparations* represent peaks and dips in the pitch trajectory which occur before and after note transitions (Saitou, Unoki, & Akagi, 2002; Maestre, Bonada, & Mayor, 2006). Especially singers use these techniques, often matched with the beginning or end of glissando segments.

2.3.2 Bowing Patterns

Bowing patterns or bowing techniques are an essential part of articulation on the violin and do not only determine the accentuation and shape of single notes, but also their transitions. The full list of bowing techniques on the violin is large and beyond the scope of this thesis. Even when ignoring the so called extended techniques, the categories used in classic violin playing cover a wide range of possibilities (Galamian, 1999, p. 74). The analysis-synthesis system presented in this thesis is not designed to cover all possible articulation or bowing techniques of the violin. It focuses on the cantabile techniques, including *Legato*, *Détaché*, *Fouetté* and *Martelé* and derived articulation styles.

Détaché, referred to as *detached* in the remainder of this work, is one of the most basic techniques. Each note is accentuated with its own stroke. A finer granulation into *simple détaché*, *accented/articulated détaché*, *détaché porté*, *portato/louré* and *détaché lancé* is not necessary in this work. The *Fouetté* is derived from the accented *détaché*, but more accentuated. The *Martelé* is a percussive stroke, with a distinct accentuation due to an increased pressure in the

attack phase. This results in a sustain part with rapidly decreasing energy, almost a supported release. Galamian (1999) describes this as the most fundamental of all strokes. It can be further categorized into *simple martelé* and *sustained martelé*.

The proposed system is less suited for faster techniques, like the Collé, the Spiccato, Sautillé and Ricochet, which are frequently used in virtuose violin play. Staccato, a series of “small, successive Martelé strokes” (Galamian, 1999) is also at the limits of the synthesis systems. Among the classic techniques not covered by the proposed synthesis system is the Pizzicato, which is not a bowed technique.

2.3.3 Glissando

Glissando refers to a transition between two notes of different fundamental frequency without a full decrease in energy and a continuous fundamental frequency trajectory. These trajectories can be described as a sigmoidal transition from the first to the second frequency, sometimes close to linear transitions. Figure 2.4 shows the fundamental frequency trajectory for the glissando transition of two-note item 22 from the sample library, a transition from A3 to D4 with vibrato in both notes.

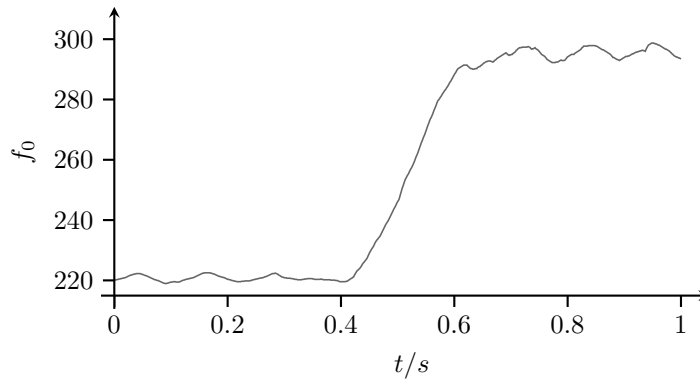


Figure 2.4: Fundamental frequency trajectory of a glissando note transition.

Models for glissando transitions have been investigated in conjunction with expressive synthesis of frequency-continuous instruments. Battey (2004) proposed the use of Bézier splines for modeling trajectories of pitch, amplitude, and spectral centroid in musical performances. All analysis was performed on singing voice, performing the *khyal* Hindustani vocal tradition. Intended for the use in musicological and compositional applications, the resulting PICACS software can be used for *expressive computer rendering*. An FM Violin and a granular sampler were used to synthesize the previously analyzed feature trajectories.

Maestre et al. (2006) use Bézier splines for modeling glissando note transitions in a system for automatic description of musical articulation gestures in singing voice performances. An experiment was conducted, showing the ability of the modeling approach to classify the articulation styles *normal*, *portamento* and *scoop*.

Ardaillon, Degottex, and Roebel (2015) use B-splines for modeling fundamental frequency trajectories in transitions as well as attack and release segments for singing voice synthesis. The

approach splits glissando transitions in two segments and introduces five knots for generating 3rd order B-spline basis vectors. This model is capable of representing preparations and overshoots and is imagined by the authors to be controlled with a simple user interface.

In the scope of this project, different parametric models for glissando modeling have been investigated in a real-time scenario (von Coler, Götz, & Lepa, 2018). The user study included the hyperbolic tangent, cubic splines and Bézier curves as models. Participants were asked to use each of these models to re-synthesize glissando trajectories from the TU-Note Violin Sample Library (von Coler, 2018), using a within subject design. The simplest model, the hyperbolic tangent, resulted in a significantly higher modeling error than the other models.

2.4 Vibrato

Vibrato refers to the periodic modulation of sound properties, including pitch, energy and timbre. Historical documents indicate that the very purpose of vibrato on the violin was to mimic the singing voice of a skilled soprano (Hauck, 2000; Hellenkemper, 2007, p. 66, p. 120). The parameters of the repetitive vibrato structure make it a timbral phenomenon, as well as a rhythmic one. Vibrato supports the individual sound of an instrument and is also an important part of expressive musical means for performers.

The specific vibrato frequency, which can vary from approximately 4 to 10 Hz, with a mean between 5 and 6 Hz (Verfaillie, Guastavino, & Depalle, 2005; H. F. Mitchell & Kenny, 2009), has an important influence on its perception. With this repetition rate, it is a phenomenon located at the boundary between *matter* and *form*, meaning that it is perceived both as a rhythmic feature – a change in sound over time, as well as a timbral quality – a perceptually stationary feature of the sound. It was in the late 1940s, when Pierre Schaeffer claimed that below lengths of 10 ms, one enters the “atomic dimension of sound”, which is beyond the “perception of form” (Schaeffer, 2012, p.44).

Just as vibrato is an ambiguous phenomenon, perceptually, the production of vibrato happens between the voluntary and involuntary actions of the performer. Fletcher (2010) points out that the mean vibrato frequency is the same as the frequency of tremors in patients with Parkinson’s disease or the frequency of repetitive light flashes known to cause epileptic fits. The vibrato frequency is not controlled directly, but by a tension in the balance of gross skeletal musculature (Westerman, 1938). This observation is the foundation for the haptic interface developed in the context of this thesis, which relies on force-sensing resistors.

2.4.1 Different Types of Vibrato

One can distinguish between three components of vibrato, namely the frequency vibrato, amplitude vibrato and the timbre vibrato, also referred to as spectral envelope modulations (Fletcher, 2001; Verfaillie et al., 2005; Fletcher, 2010). In acoustical instruments with a physical resonant body and a physical oscillation generation these types usually appear simultaneously and are in-

terconnected. However, depending on the principle of sound generation, these components are more or less dominant in specific instruments.

In most woodwind and brass instruments with a discrete frequency scale, such as clarinets, saxophones, bassoons, flutes and trumpets, the direct application of a frequency vibrato is not possible. Here, the musician can mainly modulate the intensity of excitation and an amplitude modulation is induced directly (Regnier & Peeters, 2009). Dominant spectral envelope modulations, are caused since higher intensity results in an increased relative amplitude of higher partials. Minor frequency vibrato is caused by this change in intensity, due to non-linearities and phase shifts (Fletcher, 2010).

In fretted string instruments, like the guitar, a frequency vibrato can be achieved by increasing and decreasing the string tension. This results in a lower vibrato extent than in instruments with a continuous frequency scale.

Whenever an instrument has a continuous frequency scale, a direct frequency vibrato can be applied. This accounts, for example, for the human voice. Singers use the most distinctive frequency vibrato of all instruments with a fundamental frequency deviation of 0.5 – 2 semitones for professional singers (H. F. Mitchell & Kenny, 2009).

In fretless string instruments, the musician can change the fundamental frequency directly by swaying the finger and hence changing the string's length. On the violin, this vibrato motion can be induced using three different techniques. The driving force may stem from either the arm, the hand or the finger (Galamian, 1999, p. 38). All techniques result in a clear frequency vibrato with a deviation of 0.2 – 0.35 semitones (Verfaillie et al., 2005) or up to ± 50 cents (Fletcher, 2001). Notes played in forte tend to have a more intensive vibrato than notes played in piano (Galamian, 1999, p. 38). Amplitude modulations and significant spectral envelope modulations appear as a consequence of the changing pitch.

2.4.2 Spectral Envelope Modulations

Primary spectral envelope modulations can be applied on instruments where musicians have a direct control over the properties of the resonant body. This accounts especially for the human voice, respectively the vocal tract. In other instruments, spectral envelope modulations are the result of frequency or amplitude vibrato. In the case of bowed strings, like the violin, the frequency vibrato causes shifts of the partial frequencies. The underlying frequency response of the resonant body changes the amplitudes of the partials according to their frequency. Various research projects have studied this effect for different instruments (Arroabarren, Zivanovic, Rodet, & Carlosena, 2003).

Figure 2.5 shows the spectral envelope modulations for the ninth partial from a violin sound with vibrato. Amplitude and frequency trajectories have been calculated from the second note of item 22 from the two-note sequences in the sample library. For this example, the relation between frequency and pitch is quasi linear. The higher the partial's frequency, the higher its amplitude.

Even in sounds with a significant amount of frequency vibrato, spectral envelope modulations contribute to the timbre to a great extent. Verfaillie et al. (2005) performed listening tests for evaluating the perceptual relevance of spectral envelope modulations in vibrato. An additive

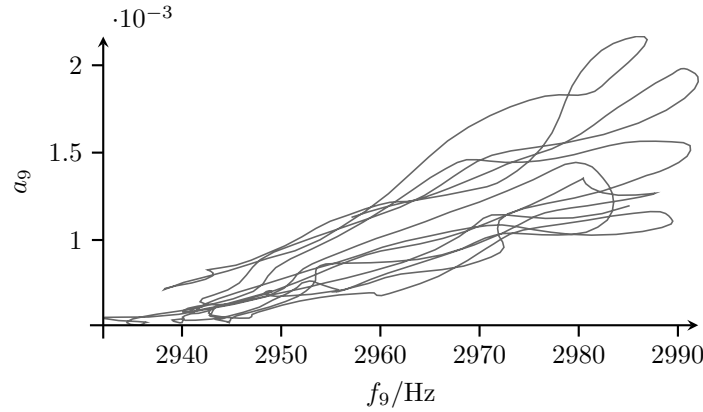


Figure 2.5: Spectral envelope modulations for the ninth partial of an excerpt from two-note item 22.

analysis-synthesis model was used which could generate stimuli with different components of vibrato. Subjects preferred synthetic versions which included the spectral envelope modulations in terms of naturalness, vibrato depth and pleasantness. Sounds with spectral envelope modulations were described to have a 'full' timbre. Studies on the perception of synthesized vibrato indicate that missing amplitude modulation and spectral envelope modulations result in lower quality sounds (Mellody & Wakefield, 2000).

2.4.3 Vibrato Analysis

Vibrato is usually considered a sinusoidal modulation of parameters. However, a vibrato model presented by Saino, Tachibana, and Kenmochi (2010) also includes the shape, respectively the waveform of the vibrato. A generalized model for a fundamental frequency trajectory $f_0(t)$ with vibrato, proposed by Pang and Yoon (2005), is a basic frequency modulation formula. Variables have been adapted to match those used in this thesis, including a steady frequency component $f_{0_{DC}}(t)$, the vibrato extent $a_{vib}(t)$, the vibrato frequency $f_{vib}(t)$ and the phase φ_{vib} :

$$f_{0_{mod}}(t) = f_{DC}(t) + a_{vib}(t) \sin(2\pi f_{vib}(t) + \varphi_{vib}) \quad (2.3)$$

Details on the above used components of the fundamental frequency trajectory are provided in Section 7.3.1. Vibrato analysis is concerned with extracting the parameters of this harmonic modulation from musical recordings. Since frequency and especially the extent of the modulation vary over time, note-related or global values are often calculated for these parameters through averaging. A general problem in the analysis of vibrato is the ratio between the length of a single note and the vibrato frequency f_{vib} . Only a few cycles fit into a note. To overcome this issue, Rossignol, Rodet, Soumagne, Colette, and Depalle (1998) propose the extrapolation of the modulation trajectory. Another problem is the modest sampling rate of the modulation signal. For standard hop-sizes of about 10 ms it yields 100 Hz. Low sampling rates lead to poor resolution in both time- and frequency domain based analysis.

Vibrato Detection

In some cases within music information retrieval it is of interest to merely detect sections with vibrato in monophonic or polyphonic music. Since vibrato is individual for specific instruments and performance styles, this information can help in many applications, such as instrument recognition, genre recognition and performance analysis. Yang, Rajab, and Chew (2017) propose a vibrato detection based on the filter diagonalization method. A Matlab-based software for graphical presentation of the results is especially useful for musicological studies. Vibrato and tremolo detection based on sinusoidal modeling has been used by Regnier and Peeters (2009) for detecting singing voice in music recordings. Amplitude and frequency modulations are calculated for individual partials. If multiple partials show vibrato during a period of time, they are assigned to a singing voice component. The interrelation between frequency and amplitude modulations, introduced in Section 2.4.1, can be used to detect the presence of vibrato (von Coler & Röbel, 2011). The advantage of this method is the minimal set of assumptions and parameters which need to be considered prior to analysis.

Time Domain Analysis

A simple and robust possibility for determining vibrato parameters is the direct analysis of the fundamental frequency trajectory in the time domain. Figure 2.6 shows an idealized modulation trajectory $f_{0_{mod}}(t)$ within a note. By detecting the local maxima and minima, frequency and extent of the modulation signal can be estimated (Prame, 1994; Rossignol, Depalle, Soumagne, Rodet, & Collette, 1999; Maestre & Gomez, 2005; Friberg et al., 2007).

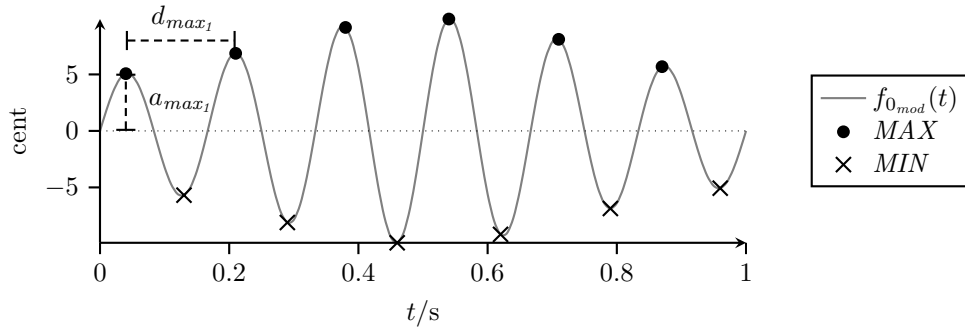


Figure 2.6: Local minima and maxima within an idealized modulation trajectory.

Based on the local extreme values, the vibrato parameters can be estimated for a note by averaging. This procedure is carried out for all $N_{max} - 1$ pairs of local maxima and all $N_{min} - 1$ pairs of local minima:

$$f_{vib} = \frac{0.5}{N_{max}-1} \sum_{n=1}^{N_{max}-1} \frac{1}{d_{max_n}} + \frac{0.5}{N_{min}-1} \sum_{n=1}^{N_{min}-1} \frac{1}{d_{min_n}} \quad (2.4)$$

$$a_{vib} = \frac{0.5}{N_{max}-1} \sum_{n=1}^{N_{max}-1} a_{max_n} + \frac{0.5}{N_{min}-1} \sum_{n=1}^{N_{min}-1} |a_{min_n}| \quad (2.5)$$

Frequency Domain Analysis

Since the vibrato waveform is mostly sinusoidal, a peak detection method in the spectral representation of the modulation trajectory is proposed in several publications (Herrera & Bonada, 1998; Arroabarren, Zivanovic, Bretos, Ezcurra, & Carlosena, 2002). For this purpose, a discrete Fourier transform of the modulation trajectory needs to be calculated. Figure 2.7 shows the spectrum of an idealized modulation trajectory $f_{0_{mod}}$. Vibrato parameters can then be estimated through peak detection.

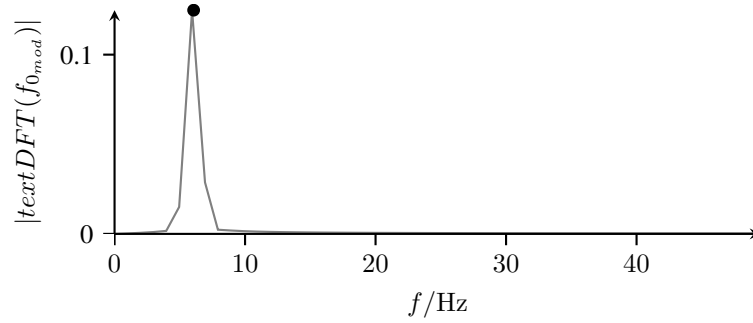


Figure 2.7: Frequency domain analysis of a vibrato modulation trajectory.

Sound Synthesis

The analysis-synthesis system presented in this thesis is a hybrid concept, combining classical sample based methods with an extended approach of spectral modeling. In order to illustrate the role of these two paradigms, Section 3.1 will give a general overview over digital synthesis approaches, their classification and characteristics. Sampling and sample-based techniques are introduced in Section 3.2, followed by a more detailed treatment of spectral modeling approaches in Section 3.3. One focus of the proposed system lies on enhancing expressive means in sound synthesis. Thus, projects dealing with expressive sound synthesis and related problems will be introduced in Section 3.4. Finally, the concept of *Spectro-Spatial Sound Synthesis* is introduced in Section 3.5.

3.1 A Taxonomy

After early synthesis concepts usually aimed at reproducing the sound of known acoustical musical instruments (Bongers, 2000), synthesizers today make up a full-fledged instrument family with different, unique paradigms of sound generation. Within this continuous evolution, novel synthesis methods have been used in order to emulate classical synthesizers, as for example in analog modeling. This development indicates that classical synthesis approaches, like the subtractive principle, earned their place as individual instruments, just as the violin or the piano.

Especially in the digital domain, a plethora of sound synthesis algorithms emerged since the computational power made their implementation possible. However, most of these algorithms are derived from a few common principles. Based on this assumption, J. O. Smith (1991) proposed a taxonomy of synthesis approaches, shown in Table 3.1. The four basic categories in this taxonomy are *Processed Recording*, *Spectral Model*, *Physical Model* and *Abstract Algorithm*. It should be noted that this list is almost 30 years old at the time this thesis is written and not fully conclusive. From a recent point of view it lacks the representation of concatenative synthesis as a technique of processed recording. Also, the now emerging concepts based on neural nets and

Table 3.1: Taxonomy of synthesis approaches proposed by J. O. Smith (1991).

Processed Recording	Spectral Model	Physical Model	Abstract Algorithm
Concrète	Wavetable F	Ruiz Strings	VCO,VCA,VCF
Wavetable T	Additive	Karplus-Strong Ext.	Some Music V
Sampling	Phase Vocoder	Waveguide	Original FM
Vector	PARSHL	Modal	Feedback FM
Granular	Sines+Noise (Serra)	Cordis-Anima	Waveshaping
Prin. Comp. T	Prin. Comp. F	Mosaic	Phase Distortion
Wavelet T	Chant		Karplus-Strong
	VOSIM		
	Risset FM Brass		
	Chowning FM Voice		
	Subtractive		
	LPC		
	Inverse FFT		
	Xenakis Line Clusters		

deep learning (Engel et al., 2017; Oord et al., 2017) should be included, either as a subcategory of abstract algorithms, or as a fifth family of synthesis approaches. Nevertheless, it serves as a point of entry and encompasses all basic synthesis approaches.

Different techniques for digital sound synthesis come with specific advantages and disadvantages and even the latter are sometimes appreciated. Artifacts from poor bit depths of early sampling devices, for example, have now become a part of the effects repertoire. Based on the taxonomy of Smith, different qualities of sound synthesis approaches have been rated by Tolonen, Välimäki, and Karjalainen (1998).

Abstract algorithms are suited for the creation of novel arbitrary sounds and very efficient emulation of known instrument sounds at a moderate quality. The digital piano or the marimbas from Yamaha’s DX7 standard presets were heavily used in the 1980s, since they sounded fairly realistic and responded well to expressive play. However, control parameters are less intuitive and usually no dedicated analysis procedure is available. Abstract algorithms are probably least concerned with this lack of expressivity, since they inherently deliver the means of mapping input parameters to changes in timbre. They represent a special case – although they too were initially also intended to emulate traditional instrument sounds, their very aesthetics made them self-contained instruments of their own.

Techniques based on processed recordings have strengths in the high quality reproduction of specific sound events. Depending on microphone, converters and playback situation, the result is indistinguishable from the original. On the downside, manipulations of sampled material require additional computational processes, leading to granular and concatenative synthesis.

Spectral approaches allow high quality analysis-resynthesis with the means for highly flexible manipulations. Depending on the model, any component of a sound can be changed individually. On the downside, the large number of operators and parameters is challenging and results in a relatively high computational cost. Another consequence is the so called *control problem*

(J. O. Smith, 1991). Meaningful metaparameters are necessary for using the many parameters of spectral modeling techniques in an expressive way.

Finally, physical models have the advantage of being intuitive, and, depending on their sophistication, computationally inexpensive. They inherently feature possibilities for expressive sound synthesis (Bonada & Serra, 2007). Through their direct representation of physical processes they offer intuitive parameters and inherent means for expressive control. They are, however, less flexible than abstract algorithms or spectral models.

As a consequence of these strengths and drawbacks, many combined synthesis approaches have been developed, trying to overcome problems and enlarge the timbral and expressive capacities. This is already the case for some representatives in Table 3.1. Thus, boundaries between different paradigms are blurred.

3.2 Processed Recording

3.2.1 Origins

On a signal processing level, *sampling* is the capturing of physical phenomena with digital systems, more precisely the digitization of analog voltages. In the musical context it represents the method of storing recorded - or sampled - sounds for playback at a later time. For classic sampling techniques this happens with a limited set of manipulations. Although John Cage already made use of records and turntables for his piece *Imaginary Landscape No. 1* in 1939, the early days of *musique concrète* can be considered the direct precursor of modern sampling synthesis. Pierre Schaeffer manipulated shellac discs and used turntables to create music with previously recorded material (Schaeffer, 2012). He composed the *Cinq études de bruits* in 1948, exploring basic techniques like manipulations in speed and the creation of loops. Another early example for a musical use of recorded material is the Egyptian Halim El-Dabh, who used a wire recorder for his experimental compositions in 1944 (Schedel, 2007). Early tape music, as for example the *Williams Mix* (1952) or the *Fontana Mix* (1958) by John Cage, makes use the same techniques as the *musique concrète* (Gurevich, 2015). Tape-based sampling techniques were also implemented in electronic instruments of the 1960s, as for example in the *Mellotron*.

Only the advent of digital means for recording and reproducing sound allowed the rapid development of sample-based synthesis. In 1971, the *Allen Organ Company* produced electronic church organs based on digital samples. From 1979 to 1981, the *Fairlight CMI*, the *Synclavier II* and the *E-mu Emulator* were released (Davies, 1996), all rather expensive and bulky devices, which were already used in many popular music productions of the early 1980s. Smaller, more affordable and released just in time, the *Linn Drum* (1982) 8-bit drum sampler can be spotted as the rhythmic foundation in various pop music tracks of the 1980s. Finally, the *Akai MPC60* (1988) implemented sampling as it is still used today and helped shaping new genres, such as hip hop and techno.

Using the appropriate recording techniques and sampling parameters, digital samplers are able to reproduce any musical sound, indistinguishable from the original. The expressive means of classical sampling, however, are poor without additional manipulations or transformations. Although single sounds can be perfectly reproduced, the concatenation of consecutive tones into a musical phrase is a problem, since the method does not implement any *prosodic rules* (J. O. Smith, 1991). Dannenberg and Derenyi state that “sampling synthesis does not offer much control over the spectral content of the signal after the initial attack” (Dannenberg & Derenyi, 1998). As a consequence, many techniques have emerged since the beginning of digital sampling, which try to add means of expressive manipulations and preserve the inherent qualities of samplers.

3.2.2 Granular Synthesis

Granular synthesis is a technique almost as old as analog sampling itself. Based on a theory proposed by Dennis Gabor in the 1940s, granular synthesis makes use of small portions of recorded sound, so called grains, to generate new sound events. These grains have typical lengths of 1 to 70 ms (Roads, 2004, p. 86). Although granular synthesis is capable of creating a large variety of

sounds, the most prominent application is the synthesis of sound textures (Dubnov, Bar-Joseph, El-Yaniv, Lischinski, & Werman, 2002).

Granular synthesis has its roots in experimental electroacoustic music. Early examples, more specifically predecessors of granular synthesis date back to the work of Iannis Xenakis in 1959 (Roads, 2006). By means of tedious tape splicing and gluing it was possible to isolate small portions of recorded sound and rearrange them arbitrarily. Although granular sequences have first appeared in his composition *Pithoprakta* (1955-56) (Solomos, 2006), the tape piece *Concret PH*, finished in 1958 is probably most prominent.

Starting with Curtis Roads (Roads, 1978), digital means brought forth several composers focusing on the use granular techniques in their work. Using the concept of micromontage (Roads, 2005), as for example in works like *Tar* (1987) for bass saxophone and tape, and *Sçir* (1988), both produced at TU Berlin, Horacio Vaggione is considered a pioneer of granular composition. Barry Truax (Truax, 1987) realized his work *Riverrun* (1986) using real-time granular synthesis on a DMX-1000 Digital Signal Processor. Various improvements, such as *pitch-synchronous granular synthesis* (PSGS) and *synchronous granular synthesis* (SGS) have been introduced (Roads, 2004) to overcome the problems related with basic concatenation of grains.

3.2.3 Concatenative

Just as granular synthesis, *concatenative synthesis* is based on the use of short elements of recorded sound, referred to as units. Originated in speech synthesis (Charpentier & Stella, 1986; Hamon, Mouline, & Charpentier, 1989), the concatenative thought is an extended granular system, using additional analysis to obtain so called *descriptors* for each unit. The original concatenative synthesis approach from the field of speech synthesis, for example, is a combination of sampling, respectively granulation and frequency domain techniques.

The concept has been adapted to musical sounds as *corpus-based concatenative synthesis* by Schwarz (2000, 2004). A corpus consists of previously recorded and segmented audio data. For each unit, a set of descriptors is calculated, which are basically spectral audio features. During synthesis, a user can now navigate the multidimensional descriptor space and thus pick samples with distinct qualities for concatenation. Although capable of working with traditional instrument recordings, this approach is especially suited for creating sound textures.

3.3 Spectral Modeling

3.3.1 Additive Synthesis

According to the Fourier theorem, any periodic signal can be expressed as a sum of sinusoids with individual amplitudes, frequencies and phases. A simplified model assumes sounds of musical instruments to be strictly harmonic. In this case, the frequencies of the individual sinusoids, referred to as partials, become integer multiples of the fundamental frequency f_0 . A finite number of N partials is considered, each with an individual instantaneous amplitude $a_i(t)$ and phase φ_i :

$$x(t) = \sum_{i=1}^N a_i(t) \cos(2\pi i f_0 t + \varphi_i) \quad (3.1)$$

Due to its simplicity and versatility, additive synthesis holds a special position in the history of electronic music and electronic sound synthesis. Additive synthesis is among the first techniques applied for electromechanic sound synthesis. Herman von Helmholtz used electrically driven tuning forks as partials for additive sound synthesis, already controlling their individual phases with resonators (von Helmholtz, 1877, p.183). The idea of generating sounds in a similar fashion has been pursued by early electromechanical musical instruments like the Telharmonium (Weidenaar, 1995), invented by Thaddeus Cahill, and the Hammond organ (Laurens, 1957). Both use rotating parts to generate sinusoidal components of additive timbres.

In the history of analog electroacoustic music, more precisely in the genesis of *Elektronische Musik*, additive synthesis was the approach which Karlheinz Stockhausen used to generate purely synthetic music (Stockhausen, 1963). When he started to explore timbre as a musical parameter to be used by composers, he imagined the assembling of complex tones by single sine waves the best way to do that, although subtractive synthesis was considered as well. *Studie I* (1953) and *Studie II* (1954) are Stockhausen's results of a purely additive approach, rigidly composing the spectral properties of musical sounds.

Finally, additive synthesis is also the original method of digital sound synthesis. In his experiments at Bell Labs in the 1950s, Max Mathews created timbres – first static, then dynamic – by superimposing single partials (Farnell, 2010, p. 267). Later joined by Jean Claude Risset and other composers, Mathews developed the MUSIC programming language (Mathews, 1969), the ancestor of most modern music programming environments. Early studies like *Daisy Bell (Bicycle Built for Two)* from 1961 already show the great potential of expressive sound synthesis by making the computer sing with formant filters.

3.3.2 Evolution of Spectral Modeling

Sinusoidal Modeling is best described as an additional higher layer for additive synthesis, expressing the behavior of partials through metaparameters. McAulay and Quatieri delivered the most prominent description of sinusoidal modeling for an application in speech synthesis and transformations (McAulay & Quatieri, 1986; Quatieri & McAulay, 1986). The resulting algorithm can be considered the basis for the following musical applications of the principle, using

only sinusoidal oscillators. A time-domain method for synthesis was proposed, attempting to preserve the phases of the original signal. The McAulay-Quatieri approach is designed to model mainly the slowly varying sinusoidal components. Fricatives, plosives or other noisy and transient parts of speech are thus not fully captured. However, using inharmonic series of a high spectral density, unvoiced segments can be synthesized using the algorithm. In a similar way, the *PARSHL* algorithm by J. O. Smith and Serra (1987) is used to model inharmonicities of sounds, as for example in the piano.

Spectral content which can be represented by a harmonic series of sinusoidal components is referred to as the *deterministic component* in the spectral modeling framework. It is also common to describe it as *tonal* or *harmonic* component. This component alone is able to capture and synthesize most pitched sounds at a decent quality, at least clearly recognizable. Yet, like speech, sounds of musical instruments contain a wide range of inharmonic signal components. Among others these include dense modes, turbulences in blown instruments, non-linear bandwidth broadening, convolutional noise in blown and bowed instruments, impulse bursts and non-linear oscillator noise (Freed, 1998).

All signal components which remain once the deterministic part x_{DET} is subtracted from the full sound x are called *residual* in spectral modeling:

$$x_{RES} = x - x_{DET} \quad (3.2)$$

Based on this decomposition, Serra and Smith (1990) propose a method entitled *Sinusoidal Modeling Synthesis (SMS)*. Within this model, the residual component is represented by a stochastic signal, respectively as time-varying filtered noise. Based on the SMS model, the harmonics + noise synthesis (HNS) system for speech synthesis is presented by Laroche, Stylianou, and Moulines (1993).

However, harmonic sinusoids and noise still can not capture all components of instrument sounds, since transient regions require a different treatment. Hence, Verma and Meng (1998), propose an extension of the *sines+noise* model with transient modeling synthesis, completing the *sines + transients + noise* model:

$$x = x_{DET} + x_{NOI} + x_{TRA} \quad (3.3)$$

Using individual modeling procedures for all three components, this full signal model is capable of high quality sound synthesis and manipulation (Verma & Meng, 1998; Levine & Smith, 1998). Since this project focuses on standard techniques of the violin, the first order residual contains mostly noise. Transients are thus not explicitly covered by the proposed model.

3.3.3 Deterministic Part

The analysis of the deterministic component is the first step in spectral modeling procedures. Trajectories of individual partials are usually obtained via a short term Fourier transform (STFT). In this process, referred to as *partial tracking*, relevant sinusoids are detected in each frame

and connected between frames. Many analysis-synthesis systems assume the sounds of musical instruments to be *strictly harmonic*. In this case, all partials are exact integer multiples of the fundamental frequency. It is well known that some instruments show significant deviations from this strict harmonic scenario. However, the violin can be considered strictly harmonic. Hence, the detection of spectral peaks can be limited to the estimated position of the harmonics. Therefore, an area around these estimated positions is defined (Serra, 1989; Hahn, 2015).

Peak heights can be detected by measuring the height of a local maximum compared to its adjacent local minima (Serra & Smith, 1990). Within the spectral frames, maxima can only be detected at support points of the Fourier transform. Once a peak has been detected, a refinement is necessary to obtain the true peak height and position. A standard solution for this is the quadratic interpolation in the Fourier transform (QIFFT) (J. O. Smith & Serra, 1987; Röbel, 2008), which is performed on the logarithmic spectrum.

Anechoic monophonic recordings, as used in the project presented in this thesis, provide ideal conditions for partial tracking. Although modeling results can still be improved, no further refinements are performed for the presented algorithm. Hahn (2015, p=21) proposes four rules for the connection of spectral peaks to partial trajectories in a similar analysis task for achieving better results. However, for polyphonic signals and noisy or reverberant recordings, partial tracking requires more sophisticated algorithms. Prediction-based approaches consider the past values of established partial trajectories (Lagrange, Marchand, & Rault, 2007, 5; Bartkowiak & Zernicki, 2011).

In sinusoidal synthesis it is possible to use the original phase relations between partials or discard them. While our perception can not distinguish different static phase relations in a mix of sinusoidal components, it is sensitive to changes in relative phase. Andersen and Jensen (2004) investigated the influence of the phase-correct synthesis of musical instrument tones with listening tests. The sound of the bass trombone was significantly degraded when discarding the original phase relations. In contrast, the Cello showed the least degradation in perceived quality, which may be transferable to the violin. The system proposed in this thesis does not use the original phases for synthesis. However, relative phase fluctuations of the partials are introduced through inharmonicity.

The initial sinusoidal model considers partials stationary within one analysis-synthesis frame. This leads to estimation errors, since amplitude- and frequency modulations of the partials occur within a single frame. The most intuitive and simple way of estimating the first order (linear) FM and AM parameters is to consider the values of amplitude and phase/frequency in the previous and succeeding frames and calculate a slope by linear interpolation (Marchand, 2012). Within an approach based on Hidden Markov Models, Depalle, Garcia, and Rodet (1993) use the frequency slope of existing partials to estimate the most likely value for the next frame. Further approaches to non-stationary sinusoidal modeling are presented by Abe and Smith (2005) and Röbel (2008).

3.3.4 Residual Part

Based on Equation 3.2, the residual is the part of an instrument's sound which remains, once all harmonic content is removed. This signal, referred to as the *first order residual* by Verma,

Levine, and Meng (1997), includes all signal components which are not captured by the harmonic analysis, namely non-harmonic sinusoids and noise. The *second order residual* contains all components which remain when the transients x_{TRA} have been removed from the first order residual. Depending on the method for estimating and subtracting the transient component, the resulting residual signal contains the actual noise, alongside modeling errors (Serra, 1997):

$$x_{RES_2} = x_{RES_1} - x_{TRA} \quad (3.4)$$

The first order residual signal can be calculated by subtraction of the deterministic part either in the time or in the frequency domain (Serra & Smith, 1990; Desainte-Catherine & Hanna, 2000). This is a critical step, since it has a significant influence on the quality of the residual signal (Rodet & Schwarz, 2007). When subtracting the tonal part from the complete signal, artifacts occur. According to Caetano, Kafentzis, Degottex, Mouchtaris, and Stylianou (2013) oscillatory behavior is left in the residual signal. Eliminating these artifacts which are correlated to the harmonic content can be achieved by smoothing the residual spectrum. Hahn and Röbel (2012) propose a method for filtering the residual in the cepstral domain. To overcome the problems by obtaining the residual through subtraction of the harmonic part, Meurisse, Hanna, and Marchand (2006) propose the estimation of the noise-distribution without the subtraction of the sinusoids. Caetano et al. (2013) evaluate the influence of different sinusoidal modeling approaches on the perception of the resulting residual. Filtered white noise was generated from an estimate of the spectral envelope, after subtracting the deterministic part. Results indicate a preference for an extended adaptive quasi-harmonic model (eaQHM) over simpler models, which consider partials either strictly harmonic or stationary within one frame.

The most common way of treating the stochastic component is to model it as white noise, processed by time-varying filters (Caetano et al., 2013). Since the phase increment of a frequency bin between two frames is random for noise signals (Serra & Smith, 1990), only the amplitude response of the filter is of interest. One common approach for is the use of auditory filter banks to obtain a compact spectral distribution of the noise. Bark-band noise modeling can be used, for example with six bark-spaced bands from 5-16 kHz (Levine & Smith, 1998). The 24 Bark bands by Zwicker (1961) are shown in Table 3.2. J. S. Smith (1999) suggests an extrapolation of the Bark scale to cover the range up to the usual sampling frequencies. Similar to the bark scale, M. Goodwin (1996) proposes the use of *equivalent rectangular bands* (ERB). Within these bands, the residual magnitude is estimated to be piece-wise constant.

For simplification, speech can be modeled as a sequence of voiced and unvoiced segments, also allowing a temporal switching between harmonic and noise. But there are fricative phonemes like /z/ which include both voiced and unvoiced components. In speech processing it is also feasible to set a separation frequency below which the signal is treated as harmonic and above which the noise components are located (Stylianou, 2001; Pantazis & Stylianou, 2008; Drugman & Dutoit, 2012).

A different method for noise synthesis in the context of expressive sound synthesis is applied in the Reconstructive Phrase Modeling framework (RPM) (Lindemann, 2007, 2010). The residual component is stored as raw audio during the analysis stage, distinguishing between transition-related noise and noise during the sustain phase, caused by excitation. During synthesis, noise

segments are picked from the corpus, leaving the noise-to-harmonic ratio as a control parameter. This method has also been applied for the synthesis of transient regions.

Table 3.2: Bark scale frequency bands with center and cutoff frequencies.

Center freq. [Hz]	Cutoff freq. [Hz]	Center freq. [Hz]	Cutoff freq. [Hz]
	20		1720
50		1850	
	100		2000
150		2150	
	200		2320
250		2500	
	300		2700
350		2900	
	400		3150
450		3400	
	510		3700
570		4000	
	630		4400
700		4800	
	770		5300
840		5800	
	920		6400
1000		7000	
	1080		7700
1170		8500	
	1270		9500
1370		10500	
	1480		12000
1600		13500	
	1720		15500

Desainte-Catherine and Hanna (2000) use probability mass functions (PMF) to model the spectral behavior of stochastic signals. Verron, Aramaki, Kronland-Martinet, and Pallone (2008) use frequency domain synthesis with noise-bands and varying window-sizes. Stylianou (2001) use a 10th order AR-Filter with a correlation based approach for modeling the noise in speech in a concatenative approach. For modeling the temporal envelope of the noise energy in speech signals, Pantazis and Stylianou (2008) compare the Hilbert transform with a moving average of the absolute values and a triangular window.

3.3.5 Window Size and Hop Size

Window size and hop size are crucial parameters for the analysis and synthesis in spectral models. As STFT parameters they have a direct impact on the quality of partial tracking and all following processes. During block-wise or overlap-add synthesis they influence not only the sound quality but also have an impact on the system's latency and responsiveness. Small window sizes are desirable, in order to fulfill the demands of stationarity within a frame. However, the smaller the window, the less low frequencies can be captured. The overlap is defined as the ratio between window size and hop size. The smaller the hop size, respectively the larger the overlap, the better the time resolution of the STFT. Yet, for large overlaps consecutive frames share the same

information, leading to a computational overhead.

According to Serra and Smith (1990), the window size should span at least four periods of the lowest fundamental frequency. Applications are found with window lengths between 20 ms (Erro, Sainz, Navas, & Hernaez, 2014) and 46.4 ms (Lagrange et al., 2007, 5) and more.

The overlap in an STFT based spectral analysis is typically between $\frac{1}{4}$ and $\frac{1}{8}$ (Hahn, 2015). For the use with a triangular window, an overlap of 50% is suggested during synthesis (Rodet & Depalle, 1992) in order to achieve a constant energy. Bonada, Serra, Amatriain, and Loscos suggest an overlap larger than 50%, in order to avoid large gains at window edges, using a Blackman-Harris window (Bonada et al., 2011, p.408). Using different window- and hop sizes for different frequency bands, Levine and Smith (1998) improve the time frequency resolution, which is a preliminary stage of the wavelet transform.

3.3.6 IFFT Synthesis

The use of the inverse Fourier transform (IFFT) for synthesis purposes has first been proposed by Chamberlin (1985) and has since then been widely used and improved (Bonada et al., 2011). Spectra are composed in the frequency domain by placing main lobes of all partials and subsequently transformed to the time domain. Noise can be generated simultaneously by defining spectral envelopes with random phases before inverse transform. IFFT synthesis has initially been introduced using an overlap-add method. Later, non-overlapping methods have emerged, since they have several advantages in representing non-stationary sinusoids (Laroche, 2000). For high amounts of sinusoidal components, the IFFT method can be significantly more efficient than time domain synthesis. According to Rodet and Depalle (1992), the IFFT synthesis outperforms the time domain method already when using 20 partials and the computational cost can be reduced by a factor of 15.

Initially, the synthesis algorithm presented in this thesis was implemented using an IFFT approach. For this application, however, the gain in performance did not outweigh the drawbacks and the increased complexity. However, aspects of this previously implemented approach will be introduced in this section.

Windows & Kernels

When performing IFFT synthesis with the overlap-add principle, individual windows can be used for the inverse Fourier transform and the overlap procedure (Rodet & Depalle, 1992). The actual IFFT window function determines the shape of the sinusoids in the spectral domain. Hence, the Fourier transform of that window should have most energy in a narrow main lobe and only minimal energy in the side-lobes (Bonada et al., 2011). In this way, only the main lobe needs to be placed with as few bins as possible. Blackman-Harris or Kaiser windows meet these requirements (Laroche, 2000).

Placing the complex main lobes requires the exact frequency and phase of the sinusoidal components. A possible way to decrease the computational cost and avoid calculating each combination is to store an up-sampled version of the window (Rodet & Depalle, 1992). In the presented real-

ization this is realized using a look-up table with *main-lobe kernels*. The space between the bins is divided into 100 equidistant steps and a kernel is calculated for each position. Each Kernel has eleven support points for representing the main lobe. Examples for different shifts are shown in Figure 3.1.

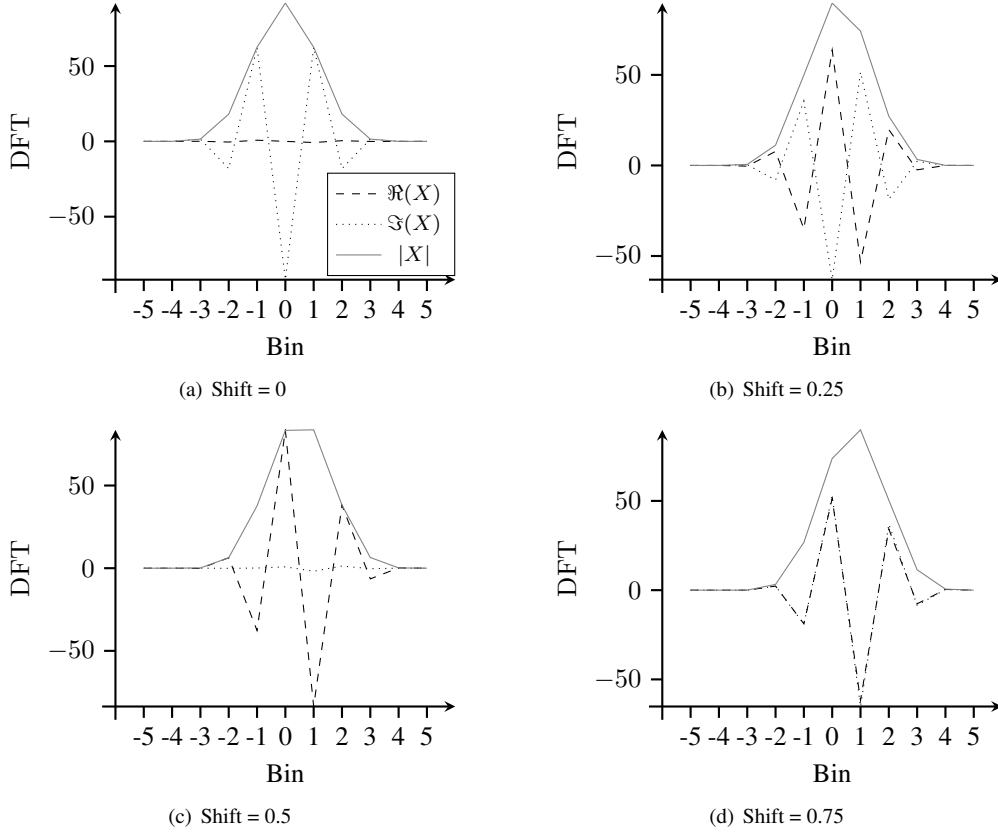


Figure 3.1: Examples for Blackman-Harris main-lobe kernels at different shifts between two neighboring frequency bins.

After IFFT and before performing the overlap-add, the temporal frame needs to be multiplied by the inverse of the IFFT window as well as by the actual overlap add window. The two windows can be combined in order to avoid two individual multiplications. A simple window for the time domain, suitable for an overlap of 50 percent, is the triangular window.

Triangular Window Phase Matching

Performing overlap-add IFFT synthesis requires the phases of a partial in consecutive frames to be matched, in order to avoid cancellations during transitions. Figure 3.2 shows a sinusoid composed of two overlapping windows without phase matching. When matched, the phase is equal at the point where both windows contribute the same amount of energy, indicated by the vertical dashed line. The condition for an overlap add without phase cancellation is that the phase Φ of the sines at frames with the indexes j and $j + 1$ is equal at the point of equal contribution (Rodet & Depalle, 1992):

$$\Phi_j[\frac{3}{4}L_{win}] \stackrel{!}{=} \Phi_{j+1}[\frac{1}{4}L_{win}] \quad (3.5)$$

For stationary frames we get the left hand side of this equation simply with the argument of the sine function at the specified point, phase-wrapped to 2π :

$$\Phi_j[\frac{3}{4}L_{win}] = \Phi_j[1] + 2\pi f_j \frac{3}{4} \frac{L_{win}}{f_s} \bmod 2\pi \quad (3.6)$$

$\Phi_{j+1}[1]$ is the phase at the beginning of the frame at index $j + 1$. It can be calculated using the phase at the critical point in the preceding frame which has been obtained in the previous step at index j :

$$\Phi_{j+1}[1] = \Phi_j[\frac{3}{4}L_{win}] - 2\pi f_{j+1} \frac{1}{4} \frac{L_{win}}{f_s} \bmod 2\pi \quad (3.7)$$

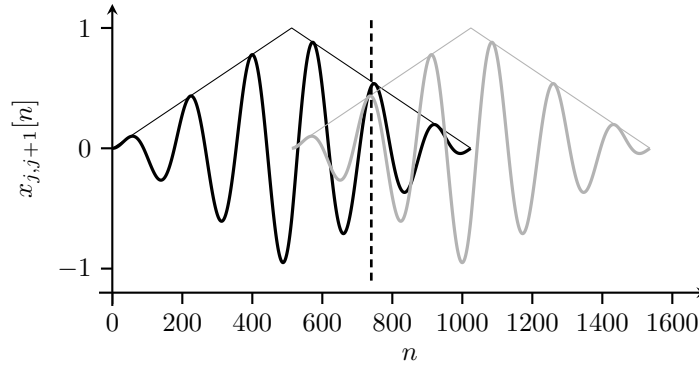


Figure 3.2: Two adjacent sinusoidal frames with a triangular window and phase matching.

Drawbacks and Alternatives

Although the gain in performance is a great advantage, there are several drawbacks to IFFT synthesis. The method as described to this point is not able to synthesize slopes in fundamental frequency within a frame, leading to staircase sounds when the fundamental frequency changes rapidly. Several improvements were presented in the past, as for example the use of chirps in the analysis and synthesis (Goodwin & Rodet, 1994; Master & Liu, 2003; Röbel, 2008). Other approaches make use of polynomial generators (George & Smith, 1992).

IFFT synthesis was finally discarded in the development of this thesis when the spectro-spatial synthesis principle was introduced. Since spectral components of the harmonic and residual part are routed to individual outputs, each component has to be isolated.

3.4 Expressive Synthesis Approaches

In the context of this thesis, the expressivity of a synthesis system, more precisely its *expressive capabilities* refers to the ability to react to control input with the desired changes in sound. Therefore, a system needs to provide *expressive control parameters* (J. O. Smith, 1991) which allow access to the timbral qualities of the synthesis process. How far the timbre can be influenced depends on the system’s means for *expressive manipulations*. In the scope of this thesis, the expressive capabilities include the application of vibrato and other modulations during the sustain, as well as articulation. Vibrato and articulation are not only important aspects of expression and musical performance. The behavior during onsets, offsets and note transitions is also an integral part of an instrument’s timbre, considered to be the *most characteristic and identifiable aspect of instrumental sounds*.“ (Lindemann, 2001).

All traditional synthesis approaches, as for example subtractive synthesis, additive synthesis, FM synthesis, sampling or physical modeling, stand on their own as means for generating sound. Yet their expressive capabilities are very different. Approaches aiming at improved expressive capabilities usually rely on the combination of different synthesis paradigms. This section contains notable examples which are based on the use of sampled sounds as well as on spectral modeling manipulations, like the system proposed within this thesis. This combination is popular since the sampling component is able to capture the timbral qualities of an instrument and allows spectral manipulations with high flexibility, especially for excitation-continuous instruments like bowed strings or the singing voice.

3.4.1 Reconstructive Phrase Modeling

Eric Lindeman developed a system for expressive sound synthesis, entitled *Reconstructive Phrase Modeling* (RPM) (Lindemann, 2007). The approach is based on manipulating and concatenating segments from expressive instrumental performances and was successfully released in the commercial product Synful (Lindemann, 2011). The software includes models for various excitation-continuous instruments, such as flute, oboe, horn, clarinets, bassoon, french horn, trumpet, trombones, tuba, violin, viola, cello and contrabass.

In the analysis stage of the RPM system, recorded phrases are manually labeled and segmented into elementary units like attacks, releases, note transitions, and note sustains. An additive signal model decomposes the deterministic part of the sounds into a slowly varying component which represents average values of single tones and rapid fluctuations of pitch, intensity and partial amplitudes (Lindemann, 2010). Rapid fluctuations of the deterministic component are stored in the *residual pitch, loudness, and harmonics + noise* (RPLHN) representation. The dependency of partial amplitudes on the control parameters pitch and intensity is modeled using a two-layer neural network with three hidden neurons. The noise component is separated from the deterministic part and stored as a pulse code modulation waveform.

During synthesis, the RPM system is controlled with MIDI messages, which generate the slowly varying pitch, loudness, and timbre trajectories. A state machine (Lindemann, 2001) selects the best matching units from the corpus, using a fuzzy search criterion. Rapid fluctuations of pitch, intensity and partial parameters, including vibrato, are taken from the RPLHN database and su-

perimposed on the slowly varying trajectories. Original vibrato is thus used from the recordings, allowing the control over vibrato frequency by varying the playback rate of the rapidly varying components. Manipulations like time stretching or compressing, pitch and intensity shifting, changes of intensity and speed of vibrato and timbre adjustments are then applied to the selected segment. Control trajectories within note transitions are directly taken from the database. Noise is added after the deterministic part has been synthesized. In violin, for example, the RPM algorithm can thus control the level of bow noise and attack noise during synthesis.

For decreasing the computational load, the RPM method uses different methods for additive synthesis in lower and higher partials. Table-lookup oscillators generate the first 20 partials, called the low-frequency harmonic sequence (LFHS). The remaining partials, the high-frequency harmonic sequence (HFHS), are coded by vector quantization in the frequency domain (Lindemann, 2010). A so called codebook of up to 150 single pitch period time-domain oscillator tables is created, using a phase randomization procedure. The upper 80 partials can hence be synthesized using one single, instead of 80 individual oscillators.

3.4.2 Spectral Concatenative

Several approaches similar to the RPM system can be summarized as *spectral concatenative synthesis*. The original concatenative synthesis (Charpentier & Stella, 1986) already performs manipulations in the frequency domain, but examples from this group combine the concatenative principle with sinusoidal or spectral models.

Perez, Bonada, Maestre, Guaus, and Blaauw (2007) present a model for expressive synthesis, trained from actual gestures from violin play. 3D-motion tracking is used to capture performance actions of performers. Bow velocity, bow pressing force, bow position and bow-bridge distance are recorded in alignment with the spectral frames of the resulting sound. The spectra are therefore divided into seven frequency bands. Neural networks are used to model the dependency of the frequency bands on the bowing parameters. Synthesis is then driven by streams of continuous performance actions. In a concatenative approach, units are selected and the neural network predicts the band energies for necessary filtering. The remaining transformations are carried out using spectral modeling techniques by Bonada and Serra (2007).

Wager et al. (2017) present a system for concatenative cross synthesis of excitation-continuous instruments. The algorithm allows the rendering of a wind instrument performance, namely a bassoon, based on control trajectories previously extracted from a string instrument performance, respectively a violin. Audio corpora, consisting of sequences for analysis and synthesis are segmented using windows of 12 ms length. For each frame, fundamental frequency, and the RMS energy are extracted as control trajectories, alongside the modulus of the windowed Short-Time Fourier Transform (STFT). By target-to-source mapping, frames from the synthesis corpus are picked according to control trajectories from the analysis corpus. Distance between two STFT frames is measured using a weighted sum of fundamental frequency and RMS to obtain close matches. A second cost function controls the smoothness of adjoined source frames by taking the Euclidean distance of the windowed STFT modulus, regarding only the areas surrounding low partials.

3.4.3 Excitation plus Resonances

Bonada, Loscos, and Kenmochi (2003) present a system for singing voice synthesis based on the *EpR (Excitation plus Resonances)* model. The system, best described as a concatenative source-filter model, generates a database from vocal performances and is able to render new performances, using the database with complementary lyric information as input. An additional user input can control expressive content of the performance synthesis.

The pseudo-physical EpR model is considering the excitation signal and the vocal tract with two cascaded filters. The source filter has an exponential decay of the partials towards high frequencies with an additional source resonance in the low frequencies, located below the first formant. The vocal tract filter consists of a vector of resonances, respectively formants, and a correction of the spectral envelope to match the original envelope. Samples, or units, are transformed with an adapted phase-locked vocoder technique, entitled *Spectral Peak Processing (SPP)*. This makes the critical separation of deterministic and stochastic component obsolete. Transposition, equalization and time-scaling are performed to match the selected units.

Based on the EpR model, Janer, Bonada, and Blaauw (2006) propose a system for *performance-driven* control of a singing voice synthesizer. Trajectories of energy and fundamental frequency are extracted from a vocal performance. Additionally, the higher level descriptors vibrato rate and vibrato depth are analyzed. For the singing voice, spectral shape is controlled via phonetic alignment, also extracted from the vocal performance. These five parameters are then used as control parameters after post-processing. An off-line synthesis engine calculates the optimal sequence of samples in order to minimize necessary transformations and performs the concatenating with the EpR model.

Performance sampling is an approach similar to the RPM system, used for singing voice synthesis based previously recorded performances (Bonada & Serra, 2007). The sonic space of an instrument-performer combination is modeled, building on the EpR model, resulting is a system capable of performance synthesis. It can be driven by a high level input, like a musical score, synthesizing performance trajectories according to previously analyzed performances. The performance trajectories include the phonetic unit sequences and the pitch and loudness envelopes. Best matching samples, respectively units, are then selected from the corpus and transformed for optimal fit and smooth concatenation. Amplitude and phase envelopes of the VPM spectrum (Bonada, 2004) are considered, as well as the controls of each formant in the EpR model. Spectral envelope modulations are also captured and reproduced by this approach, reported to deliver high quality results.

3.4.4 Spectral Interpolation Synthesis

A system presented by Dannenberg and Derenyi (1998) creates instrument models by analysis of solo performances, in a way similar to the algorithm proposed in this thesis. A phase-vocoder-based algorithm (Beauchamp, 1993) extracts spectra, RMS, and fundamental frequency, as well as partial amplitudes for each pitch period of the complete recording. For a trumpet model, a maximum of 30 partials is stored. These harmonic spectra are represented by an array of

relative partial amplitudes and arranged in a two-dimensional matrix indexed by a combination of fundamental frequency and RMS. Original phases are discarded and thus not synthesized. Attack portions of the sounds are stored as raw waveforms.

During synthesis, the system is controlled with the modulation sources RMS amplitude and fundamental frequency. Using a framerate of 20 Hz, the four nearest spectra of the instantaneous amplitude-frequency combination are selected and interpolated. One period of the timbre described in the harmonic spectrum is generated in the time domain, resampled and amplified to match the desired amplitude and frequency. Unpublished listening tests showed that the original phases of the harmonics are not necessary for a perceptually convincing synthesis. However, the lack of representing inharmonicity reportedly deteriorates the results for instruments like trumpets, trombones, and saxophones.

3.4.5 Removing the Time Axis

D. Wessel, Drame, and Wright (1998) present an approach for *removing the time axis* in sinusoidal modeling, comparing a parametric and a non-parametric machine learning model. For this, solo performances of excitation continuous instruments like the voice, wind instruments, or strings are analyzed using sinusoidal modeling. Training data is calculated in advance, storing pitch, loudness and brightness as control parameters alongside the spectral frames. Machine learning is applied for modeling the relationship between the control parameters fundamental frequency and loudness and the sinusoidal parameters, respectively the spectral content. The algorithms are capable of rendering audio from novel control trajectories, reported to result in convincing *global timbral instrument behavior*.

The parametric approach is a feed-forward neural network, consisting of 80 hidden layers and output units, trained using a back-propagation learning method. After training the network, partial trajectories can be generated from the input parameters loudness and fundamental frequency. The non-parametric memory-based approach stores a full training set of control data and associated spectral frames in matrices. During synthesis, for each input triplet of pitch, loudness, and brightness, the algorithm is searching for the closest triplet in the memory.

Experiments included the sufling flute, saxophone, voice and viola, respectively viola glissandi. The parametric model is more compact, can be generalized and provides smoother sounding results.

3.4.6 Extended Source-Filter Model

Source-filter models have their origin in speech synthesis and processing and were already explored at an early stage in the analog domain (Dudley, 1939). In the digital domain, source-filter models have also been used for the synthesis of musical instrument sounds. Similar to the approach presented in this thesis, a model combining sample libraries of single instrument sounds with spectral modeling is presented by Hahn, Röbel, Burred, and Weinzierl (2010). A sample library captures the whole pitch range of an instrument at three different intensity levels. Deterministic and stochastic spectral components of the sounds in the library are modeled using an extension of the source-filter-decay approach proposed by Klapuri (2007). As proposed by

Burred, Röbel, and Rodet (2006), a distinction is made between *f0-correlated features*, such as the basic waveform, including for example the odd to even ratio and the *f0-independent features*, including formants and resonances, respectively the features of the resonant body. The *f0*-correlated features are included in the source signal, whereas the *f0*-independent features are included in the filter properties.

Within the model, the variation of the spectral envelope is assumed to be directly related to the relative energy level of the signal (Hahn et al., 2010). It is further assumed to be constant for specific relative energy levels but individual for different levels. It thus changes the instrument's timbre according to the intensity input. The behavior of each partial in dependency of the relative signal level is encoded in the source model using superposed B-Spline polynomials, using a different models for the *attack-to-sustain* and the *sustain-to-release* (Hahn et al., 2010). An instrument sound is thus segmented into two overlapping portions. B-Splines are used for modeling the filter. The time-varying spectral envelope depends on a linear combination of the oscillator model and the filter model.

Based on the above introduced source-filter model, an extended version is presented by Hahn and Röbel (2012), including the representation of the residual component and other refinements. Four control parameters, namely pitch, global intensity, local intensity and temporal segmentation are defined during the analysis stage. Noise and harmonic content are separated in the analysis process and modeled individually.

In contrast to its predecessor, the extended source-filter approach also models the variations of the spectrum related to the global intensity in the source (Hahn, 2015). The B-Spline approach is extended to the three dimensions pitch, global intensity and local intensity using tensor products. For both the attack-sustain and the sustain-attack segment, an individual hyperplane models the temporal behavior of each individual partial. The filter, respectively the resonator, remains constant and is again modeled using B-Splines.

Cepstral coefficients of the residual, here referred to as the noise signal, are modeled with the same B-spline tensor product. Both for the attack-sustain and the sustain-attack segment, each cepstral coefficient is expressed as a function of pitch, global intensity and local intensity. The parameters of the model are trained from a data set for noise and harmonic component, separately by linear regression. Henrik Hahn limits his expressive synthesis approach for the violin to the parameters pitch and intensity (Hahn, 2015, pp. 54 f). This model is extended to four parameters (Hahn & Röbel, 2012), namely *Pitch*, *Global Intensity*, *Local Intensity* and *Temporal Segmentation*.

3.4.7 Statistical Parametric Speech Synthesis

Hidden Markov Models (HMM) have been frequently used in systems for statistical parametric speech synthesis, an alternative approach to the established unit selection algorithms in speech synthesis (Zen, Tokuda, & Black, 2009). The generative models are used to model the behavior of an excitation signal and spectral features in dependency of a target word sequence. More recently, the approach is further improved by using generative adversarial networks (GANs) (Saito, Takamichi, & Saruwatari, 2017).

3.5 Spatial Sound Synthesis

3.5.1 Spatialization & Diffusion

In the field of electronic and electroacoustic music, *spatialization* is the process of distributing sound or music on sound reproduction systems of any configuration. A reproduction system is any electroacoustic setup for generating sound from electrical signals, including a single loudspeaker, headphones, stereo systems, quadraphonic systems and other surrounding setups or arrays of hundreds of speakers. The practice of spatialization is an integral element of electronic and electroacoustic music and has been part of the compositional means and performance practices since the beginnings. Although already considered in acoustic music, space became graspable as a dynamic musical parameter by electronic means, and has hence been used as such. With the concept of *Spatiomorphology*, Denis Smalley Smalley (1997) delivers a theoretical framework for describing the spatial aspects of acousmatic music.

Even in early monophonic tape compositions, like *Symphonie pour un Homme Seul*, composed by Pierre Schaeffer and Pierre Henry from 1949 to 1950, the dynamic use of amplitude envelopes and the control of direct to reverb ratio can be considered a form of spatialization. At this point Schaeffer already saw the possibility and the need to use the dimension of space as a performance parameter when playing recorded music to an audience (Battier, 2015). As a consequence, *sound diffusion*, or simply *diffusion*, the practice of spatializing pre-composed music on loudspeaker systems developed into a standard technique for the presentation of electroacoustic music. Dennis Smalley describes it as “*the projection and the spreading of sound in an acoustic space for a group of listeners*” (Austin, 2000). Originally, diffusion is used to distribute stereo signals on loudspeaker systems with a larger number of speakers, dynamically adjusting it to the acoustics of the performance space and supporting the musical gestures. However, *multichannel diffusion* (Wilson & Harrison, 2010), the individual distribution of multiple audio streams emerged as a method in the performance of electroacoustic music.

On reproduction setups with two or more loudspeakers, spatialization commonly refers to the positioning of sound through means of panning or sound field synthesis methods. Even stereo, the dominant format in electroacoustic music on tape, allows the creation a sound image with width and depth behind the speakers (Harrison, 1998). However, most progress and evolution in spatial composition and performance happened on quadraphonic and octaphonic loudspeaker setups, surrounding the listener in the horizontal plane. This was already the principle behind early electronic compositions like *Kontakte*, composed from 1959 to 1969 by Karlheinz Stockhausen (Stockhausen, 1971, p.28). Although Stockhausen refers to his composition *Gesang der Jünglinge* from 1956 as the first one ever to allow musicians to control the movement of sound in the room (Stockhausen, 1964, p.40), the electronic version of *Kontakte* makes use of the quadraphonic setup for sound movements in the two-dimensional plane.

As with sound synthesis and other areas of electroacoustic music, spatialization of music made a leap with the rise of digital signal processing. Mostly acknowledged for the invention FM synthesis, the work of John Chowning stands out as ground breaking in the use of digital means for spatialization. For his compositions *Sabelithe* (1971) *Turenas* (1972), he implemented a dynamic

sound-object based panning algorithm, enhanced with Doppler shift and digital reverberation in *Music IV/Music 10* (Chowning, 2011). This paradigm of moving virtual sound sources in panning or sound field synthesis systems remained the key concept in most recent spatialization systems and still is today. Single sounds are treated as point sources, in an *object-oriented* way, mimicking the way sounds of actual physical sources behave (Hagan, 2017). Different technologies like Vector Base Amplitude Panning (VBAP), Distance based amplitude panning (DBAP), Ambisonics or Wave Field Synthesis (WFS) can be used for this approach (M. Baalman, 2010). Trajectories of virtual sound objects can be programmed or defined during composition, driven by algorithms or controlled with during performance using gestural interfaces.

Spatialization has moved on, making use of various techniques to create an increased sense of envelopment or engulfment (Lynch & Sazdov, 2011), by spatial decorrelation and other means. Instead of just panning sound events and streams or placing them as virtual point sources, a spatial image is created by means of spectral processing, granulation and others.

3.5.2 Spectral Spatialization

Spectral spatialization represents a special case of spatialization. Single sounds or audio streams are decomposed into frequency components which can then be treated individually during spatialization. Frequency components can be extracted from material in the production process or in real-time during the performance. This principle is also used in mixing and production of conventional stereo music. DAW plugins like *Waves PS22* allow the individual panning of frequency bands to enhance the stereo image (James, 2012).

The splitting of audio streams into frequency bands for an individual spatialization is a common practice in electroacoustic music performances. Normandeau (2009) introduces the term *timbre spatialization* for the individual spatialization of a signal's frequency components. In his electroacoustic works he makes use of a filter bank using four dynamic band-pass filters, initially using discrete loudspeaker signals, then moving on to Vector Base Amplitude Panning (VBAP). Various approaches for *timbral spatialization* are presented by Schmele (2011). "Artificially constructed spatial sounds are usually groups of individual point sources." (Schmele, 2011, p. 98) In the case of spectral separation, spatial movement of the sound image can be achieved by fluctuations of the spectral content, even when leaving the virtual sound sources in position. This technique can be traced back to early systems for sound diffusion, like the *Acousmonium* at Groupe de Recherches Musicales (GRM), where an inhomogeneous set of loudspeakers is used to allow spectral spatialization (Wilson & Harrison, 2010).

More specific treatment of spectral components for spatialization can be performed in the frequency domain, after Fourier transform. A common technique are *spectral delays*, where FFT bins are delayed and spatialized individually (Kim-Boyle, 2008). A system for panning single FFT bins in stereo and quadrasonic setups, realized in *Max/MSP* is presented by Torchia and Lippe (2004). Control is realized using a *multiSlider* slider GUI object in *Max/MSP* and automation patterns. Based on this work, Kim-Boyle (2005) uses the Boids algorithm for an individual spatialization of FFT bins.

James (2012) extends *Wave Terrain Synthesis*, a 2- and 3-dimensional wavetable approach for

sound synthesis, to the spatial domain. Signals are therefore split into 1024 frequency bands by Fourier transform. The virtual position of each band, more precisely each bin, within any 2-dimensional multi-channel speaker configuration is then determined by terrain a contour. Rather than creating the impression of spatial movement, this kind of spectral spatialization results in a complete sound scene. James defines this as *timbre spatialization*, the combined exploration of spectromorphology and spatiomorphology (James, 2015). A small number of control parameters is needed to shape the wave terrain and hence the spectral distribution, since the terrains are generative (Hope & James, 2013). Through mapping, the global distribution of all bins is controlled, jointly. Control in the system is inspired by graphical synthesis systems like Iannis Xenakis' Upic. Trajectories of are used to scan the wave terrain, either drawn, generated by algorithms or extracted from other data.

Spectral spatialization is not only applied in electroacoustic music, but also in the field of virtual acoustics. One application for is the synthesis of radiation patterns in virtual acoustics. A system with three cubical loudspeaker arrays is used to synthesize radiation patterns in three frequency bands, weighted by the first order spherical harmonics (Dirogis, Warusfel, & Caussi, 1996; Misdariis, Warusfel, & Causse, 2001; Warusfel & Misdariis, 2001). Böhlke and Ziemer (2017) measure violin radiation patterns with 128 microphones arranged in a sphere. Individual patterns are calculated for 13 relevant bands in the Bark frequency scale. 128 virtual monopole sources are used within a WFS systems to synthesize the violin sounds with their radiation patterns. Listening tests revealed that this procedure results in a degraded, less natural violin sound with additional comb filter sensations and other artifacts, caused by the massive processing.

Virtual reality and gaming contexts are a major application for spatialization. Pulkki, Laitinen, and Erkut (2009) present an approach for increasing the spatial extent of virtual sound sources in such immersive environments, based on Directional Audio Coding (DirAC). It aims at sound textures, such as flocks of birds, breaking waves, steps of multiple persons or applause. The angle of incidence for a monophonic source is modulated randomly within given constraints, determined by the desired source width. This modulation happens frequency-dependent, for frequency bands of an equivalent rectangular bandwidth (ERB) filter, and can also be extended to create sound sources enveloping the listener. The technique has been evaluated for different sound sources and refined Pihlajamäki, Santala, and Pulkki (2014).

3.5.3 Spatial Sound Synthesis

Spatialization as a means for processing and manipulating sound sources is in most cases used as an audio effect. However, sound synthesis can be carried out with an inherent spatial aspect, resulting in a tight link between the sound spatial and its spatial properties. Schumacher and Bresson therefore define spatial sound synthesis as “*spatialisation processes at the micro-level of sound*” (Schumacher & Bresson, 2010, p. 11). Depending of the synthesis paradigm, the spatialisation is introduced at a stage where the single components are still unfused, allowing to shape sound spatially. The boundaries between spatialization and spatial sound synthesis are blurred. As with most audio effects, extreme parameter settings in spatialization can result in completely altered and novel sounds.

Within the context of acousmatic music and diffusion, James (2015) distinguishes between *point-source panning*, respectively the control of virtual sound source positions and the control over the *spatial extent*. It is common to use groups of individual point sources to create synthetic spatial sounds (Schmele, 2011; Hope & James, 2013), which is the continuation of the principles introduced by John Chowning for digital spatialization techniques. Single audio streams are routed to individual virtual sound sources in a sound reproduction system based on VBAP, WFS, Ambisonics or binaural synthesis. Properties like source position, extent and radiation patterns of these virtual sound sources can be then controlled. This is referred to as *object-based spatialization* (Perez-Lopez, 2015), *source-centered* (Garcia, Carpentier, & Bresson, 2017) or *point-source* approach. This paradigm is well suited for spatializing both granular synthesis and spectral modeling, which are the most common techniques in spatial sound synthesis. In addition it is graspable through its meaningful parameters and agnostic of the reproduction system. Alternatively, *channel-based* (Perez-Lopez, 2015) approaches, also referred to as *speaker-centered* (Garcia et al., 2017) or *immersive techniques* (James, 2015), control the contribution of single loudspeakers to the rendering of an audio stream through more abstract means. This liberation from the restrictive point-source paradigm offers novel, more progressive possibilities but is harder to formalize and less portable, since it depends on the loudspeaker setup.

A central aspect of spatial sound synthesis, both point-source oriented or channel based, is the possibility to make the fusion and separation of a sound and its components an accessible parameter of composition and performance. These poles are also described as *unity and multiplicity* (Topper, Burtner, & Serafin, 2002) or as *integration and segregation* (Nyström, 2015) and have an influence on the perceived *objecthood* of synthetic sound.

Granular Approaches

Granular synthesis and related methods offer an intuitive approach to spatial sound synthesis. Due to the inherent segmentation of the audio stream, every segment, respectively every grain, can be spatialized individually. Spatial granular synthesis results in remarkable effects, since the grains have a transient character and are hence easy to locate for the auditory system.

Although working with discrete signals for eight loudspeakers, some of these possibilities can already be perceived in Cage's *Williams Mix* (see p.32). A dense spatial sound texture is created by simple means. Yiannis Xenakis' early granular composition *Concret PH* for the Philips Pavilion at the 1958 World Fair in Brussels was specifically composed for spatialization (Valle, Tazelaar, & Lombardo, 2010). Loudspeakers in the pavilion were organized in sound routes, allowing the granular structure to move through the listening space.

In the digital era the spatial possibilities continued to be an important aspect of granular synthesis. In his book *Microsound*, Curtis Roads outlines several aspects of spacial granular synthesis. *Scattering* is achieved by assigning "*an independent spatial position to every sonic particle*" (Roads, 2004, p.223), creating a three-dimensional sound picture. Depending on grain density and movement, complex sound objects can be created in this way. *Per-Grain reverberation* is introduced as a means for deepening the spatial image. Every grain can be synthesized with an individual direct-to-reverb ratio, creating individual senses of distance for the grains or even with individual reverb characteristics. Other methods include modulation or convolution of input

signals with spatialized grains.

Barry Truax makes use of the *DM-8* system, developed at Simon Fraser University, for granular spatialization. It allows the rendering of eight discrete channels, as well as dynamic trajectories for panning. Truax combines granular processing with Karplus-Strong-based (Karplus & Strong, 1983) resonators for spatialization in his compositions from the 1990s. At that point, according to Truax, the composition of sound and its internal timbral properties, its *volume*, was still separated from the spatial composition. He saw the full integration to a spatial synthesis system the next step, possible through means of DSP technology.

Kim-Boyle (2005, 2008) uses *Jitter* objects in *Max/MSP* to generate particle systems for spatializing grain-based synthesis. Simple panning is applied to single grains, following trajectories by different algorithms on a three-dimensional reproduction system with five loudspeakers. The *Boids* algorithm calculates trajectories, based on flocking behaviour of birds. Grains are transposed to create a synthetic Doppler effect.

McLeran, Roads, Sturm, and Shynk (2008) use granular, respectively dictionary-based methods in stereo and two-dimensional loudspeaker setups within a custom software *Scatter*. *Scattering* is introduced as a fully stochastic panning of each single grain, *blurring* as a mild scattering with a perceivable center. Transitions between these modes are introduced as *convergence and divergence*. By parametric filtering, atoms with specific properties can be assigned to spatial centers, resulting in a spatial distribution which is tightly linked to the timbral characteristics.

The *BMSwarmGranulator*, part of the *BEASTMulch* project, relies on the Boids algorithm for spatializing granular sound textures (Wilson, 2008). Streams of grains move in two- or three-dimensional trajectories defined by swarm behavior. In contrast to the predominant point source panning approaches, the SuperCollider extension uses hard assignment. For each grain, the algorithm calculates the nearest loudspeaker at the spawn time and uses only this speaker for rendering it. This method is in the line of the *BEAST* paradigm, using a larger number of more than 90 loudspeakers with individual characteristics. The result is intended to be a *diffuse but localized sound*, rather than individually perceivable grains.

Closely related to granular synthesis, Einbond and Schwarz (2010) make use of corpus-based concatenative synthesis (Schwarz, 2000) for spatial sound synthesis. The spatial arrangement of the units, respectively their descriptor coordinates, are directly mapped to the two or three-dimensional space of a Wave Field Synthesis (WFS) or Vector Base Amplitude Panning (VBAP) system. When navigating through the corpus during performance, the spatial position of the units is directly derived from the descriptor coordinates with an additional scaling parameter. This method synthesizes spatial sound textures with a deep coupling between timbral and spatial characteristics. A method for granular processing of spatial recorded sound has been proposed by Rosli and Roads (2016). The first order spherical harmonics are captured using a *Soundfield ST350* surround microphone. Grains are then created in the temporal, as well as in the spatial domain.

Abstract Algorithms

Principles of abstract algorithms can also be used for spatial sound synthesis. A system for modulation synthesis, considering both amplitude modulation (AM) and frequency modulation (FM) on 2D and 3D loudspeaker systems is presented by McGee (2015). The goal is to create a complete instrument which unifies space and timbre. Instead of directly modulating the frequency as in conventional FM synthesis, the Doppler effect is utilized. Using extreme virtual source velocities, *Doppler FM* is thus realized, where the speed of the sound source is proportional to the frequency and depth of the frequency modulation. Typical FM timbres start to occur at velocities of about 125 m/s. Spatial modulation changes the source position and thus the amplitude through gain attenuation with increasing distance, resulting in spatial AM. The spatial modulation technique is implemented as a VST plugin, including Distance-Based Amplitude Panning (DBAP), Vector Base Amplitude Panning (VBAP) and Ambisonics, allowing the timeline automation of all relevant parameters.

Physical Models

Physical models are designed to represent actual physical instruments and thus have an inherent virtual spatial configuration, a topology and even a geometry. Mapping these configurations and geometries to systems for spatial sound synthesis is thus obvious. In the context of *Spatio-Operational Spectral Synthesis* (SOS), Topper et al. (2002) apply physical models in multichannel loudspeaker systems. Friction force and velocity of a waveguide-based string model are separated and sent to two individual loudspeakers. The result was a splitting of the sound into two separate objects. Another experiment was based on a physical model of a singing bowl, described in detail in the following paragraph. Implemented in both MAX/MSP and RTcmix, trajectories in SOS can be controlled with a graphical user interface.

Based on the SOS framework, Burns, Serafin, and Burtner (2003) create networks of eight individual digital waveguides with different interconnections, resulting in circular and spoke topologies. The waveguides are tuned to represent resonant modes of a Tibetan singing bowl, excited with recorded attack transients. Each waveguide is then tapped with an outlet and all eight outlets, each representing a resonant frequency, are sent to a spatialization system. The simplest case assigns each waveguide to a loudspeaker in an eight channel system. By dynamically distributing the components of the sound in space, the objecthood of the model can be extended and the conference of the sound becomes a musical parameter.

Physical models have also been combined with Wave Field Synthesis (WFS) for spatial sound synthesis by Müller and Rabenstein (2009). A virtual string is realized using the Functional Transformation Method (Rabenstein & Trautmann, 2003). Directly connected to the string model is a virtual piston model as a sounding board, approximating the radiation pattern in a simplified way. This radiation pattern is then used to calculate the loudspeaker driving functions of the WFS system. The approach has also been adapted for two-dimensional membrane models (Rabenstein, 2010).

Topographic Synthesis

An approach for synthesizing spatial sound textures beyond the conventional point-source paradigm is presented by Nyström (2018). *Topographic synthesis* uses multiple single-channel synthesis processes and assigns each to a loudspeaker in a larger reproduction system. This can be implemented with any known synthesis approach, including spectral and granular techniques. Initial examples are noise-fed filterbanks and additive synthesis through oscillator banks. The spatial texture is then controlled by parameter distributions across all instances, respectively all speakers. Linear and non-linear parameter envelopes, realized in SuperCollider, are used to control the distribution of a given parameter on all speakers. Results of the topographic synthesis are not intended to be perceived and localized as virtual sound objects but in shifts of textural qualities, allowing both the integration and segregation. This concept bears resemblance to the concept of timbre spatialization. Yet it is clearly designed as a system for sound synthesis, fusing timbral and spatial properties.

3.5.4 Spectro-Spatial Sound Synthesis

For the scope of this thesis, *Spectro-Spatial Sound Synthesis* is defined as the spatialization on the synthesis level, using additive synthesis and sinusoidal or spectral modeling. In contrast to spectral spatialization, are actually generated with an inherent spatial structure. Although this seems to be an obvious technique and spectral spatialization is common in electroacoustic music, it gained less attention than grain-based approaches. This is probably due to the less dominant effect of spatialization when working with sinusoidal or other spectral components, as compared to grains with transient character. Responses of participants in the user study, presented in Section 10.5.2, also confirm this.

In his book on auditory scene analysis, Albert Bregman addresses the aspect of *spatial correspondence* (Bregman, 1990, p.293–311). Acoustic components localized at the same origin are likely to be assigned to the same auditory stream or sound source. On the other hand, spectral components of a sound are still fused when distributed spatially. Bregman conducted compositional experiments with spatialized additive synthesis, focusing on this phenomenon. The sound of an oboe was split into groups of odd and even harmonics which were then reproduced on individual loudspeakers, left and right of the listener. The result is perceived as a phantom source, centered between the two speakers. When frequencies of the harmonics on the left and right side were modulated with uncorrelated signals, the integration fell apart. This, however, would also happen without spatial splitting. Frequency modulation with correlated signals resulted in an integrated sound, again. Based on our experience, the auditory system tries to assign all components of a sound to the same origin. This has an influence on the use of spectral spatialization, since this ambiguity can be controlled by changing the spectral and spatial properties jointly or decoupled.

Freed (1998) applies spatial sound synthesis in a framework for IFFT synthesis, as introduced in Section 3.3.6. Two independent spectra are generated from sinusoidal kernels in the frequency domain which is more performant than creating two individual spectra. This allows a component-wise stereo panning, useful for a spectral decorrelation. The authors also point out the possible application of the approach in the dynamic synthesis of radiation patterns in instrument resyn-

thesis.

In the context of *Spatio-Operational Spectral Synthesis* (SOS), Topper et al. (2002) conducted a series of experiments with additive synthesis on circular eight-channel loudspeaker setups. Initially, the first eight harmonics of a square wave and sawtooth were statically assigned to an individual speaker, resulting in a predominant effect of fusion. For the next stage, each harmonic was assigned a circular path on the loudspeaker system by panning. Trajectories were delayed by one speaker per harmonic, resulting in an ambiguity between *oneness and multiplicity*, referred to as the *SOS effect*.

Schumacher and Bresson (2010), Bresson (2012) present a system for using spatial sound synthesis in the compositional process, based on *Csound* and *OpenMusic*. The *OMChroma* framework is used for sound synthesis, including classes for additive and granular synthesis, FM synthesis and other algorithms. This synthesis stage is extended with *OMPrisma*, a *Csound* framework implementing a variety of spacial sound rendering techniques, such as VBAP, RVBAP, DBAP and Ambisonics. Trajectories can be defined using B-Splines, 2D and 3D breakpoint curves and stochastic processes and edited directly. They are then used to control both the synthesis and the spatial aspects. The *OMChroma/OMPrisma* toolset is intended to be used in a modular fashion, allowing different combinations of synthesis and spatialization algorithms. Example applications include polyphonic spatialization of plucked string algorithms as well as granular techniques and *spatial additive synthesis*, the synthesis of hundreds of partials with individual spatialization parameters.

Besides musical applications, gaming and virtual reality systems are an interesting field for spatial sound synthesis. Verron et al. (2008) present a method for spatial additive sound synthesis, focusing on environmental sounds. Using the IFFT approach, single sounds of an auditory scene are composed as series of short term spectra, which are spatially encoded. All sound sources are then added up and a signal is synthesized for each loudspeaker with IFFT and overlap-add. Hence, this approach has a significant advantage over time-domain methods for a large number of sources. Virtual sound sources with spatial extent can be created by synthesizing multiple decorrelated point sources from one signal. For additive synthesis, splitting one sound source into multiple point sources, each with individual spectral portions of the full sound is suggested in the publication. A maximum of eight point sources is arranged on a circle around the listener, each with a decorrelated version of the original signal (Verron, Aramaki, Kronland-Martinet, & Pallone, 2009). The source width can then be controlled by controlling the angle spanned between the point sources. Control over the synthesizer is organized according to the categories *solids*, *liquids*, *air* and *explosion* of environmental sounds (Verron, Pallone, Aramaki, & Kronland-Martinet, 2009). Sounds of each category have their own set of high level textural and spatial parameters.

Control & Mapping

At the beginning, Section 4.1 explains the role of the interface in a digital musical instrument in general. The project presented in this thesis involves the combined control of sound synthesis and spatialization. This direct relation has only been sparsely investigated. Background and related concepts for the control of sound synthesis and of spatial control are thus presented independently in Section 4.2, respectively Section 4.3. Section 4.4 introduces the concept of mapping in the context of digital musical instruments, focusing on the practice of user-defined mapping.

4.1 Interfaces in Digital Musical Instruments

The design and exploration of user interfaces is an integral part of the design of electronic musical instruments. As following sections point out, this accounts for the control of sound synthesizers, as well as for the interaction with spatial audio systems. Due to the growing need for a focused scientific discourse, the conference on *New Interfaces for Musical Expression* (NIME) emerged from the more general *Conference on Human Factors in Computing Systems* (CHI) in the early 2000s. Since then, most of the research relevant to this topic originates from the NIME community.

Within a comparison between musical interfaces and general human computer interaction, Wanderley and Depalle (2004) present seven contexts of interfaces in interactive computer music. Among these, *musical instrument manipulation* and *sound processing control* are of interest for the scope of this thesis. Referring mainly to digital instrument manipulation, four features are listed to be of special relevance: *Learnability*, describing how easy it is to learn a controller; *Explorability*, referring to the variety of gestural possibilities and the associated mapping; *Feature Controllability*, including the relationship between gestures and changes in the resulting sound; *Timing Controllability*, referring especially to precise timing then to the rapid execution of tasks. All aspects of control treated in this section have an effect on these contexts.

Sensors are the first link in the design of digital musical instruments, since they turn physical phenomena into utilizable data streams. In this context we focus on muscle-driven sensors or transducers, since they offer the most expressive access. Other concepts, based for example on psychophysiological measurands like skin conductance or brain activity through electroencephalography (EEG), are generally not considered in the design of expressive interfaces. Several attempts were made to find a taxonomy for sensors in the context of DMI. Based on human computer interaction research by Mackinlay, Card, and Robertson (1990), a categorization of transducers for musical applications is presented by Vertegaal, Ungvary, and Kieslinger (1996):

- position sensors (e.g. fader, touch display)
- rotary position sensors (e.g. modulation wheel, rotary potentiometer)
- velocity sensors (e.g. computer mouse)
- rotary velocity sensors (e.g. rotary encoder)
- isometric force sensors (e.g. aftertouch, FSR)
- isotonic force sensors (e.g. accelerometer)
- isometric rotary force sensors
- isotonic rotary force sensors (e.g. pitch bend)

Most important is the distinction between isometric sensors, which do not require motion but merely the application of force, and isotonic sensors, which require motion. These modes are suitable for different purposes. As stated in Section 2.4, for example, vibrato is connected with a general tension in the skeletal musculature. Purely isotonic sensors do not enable such a tension.

In a thorough systematic survey, Bongers (2000) proposes a framework for classifying interaction in digital musical instruments, with a focus on sensor types and their application in real-time musical projects. Within this framework, muscle-based actions are categorized as shown in Figure 4.1. Force-sensing resistors and switches are mentioned as isometric sensors, which have both been integrated into musical interfaces in various ways. Isotonic sensors are further divided into those based on mechanical contact and contactless principles. The first category includes rotation potentiometers and encoders, joysticks, pads and ribbons, touch pads and bend sensors. Contactless sensors allow mid-air interaction, for example based on magnetic field sensors, accelerometers, gyroscopes and photoresistors. Camera-based tracking systems are not mentioned but also belong to this category.

Depending on the individual application, all sensor types can be considered when designing interfaces for digital musical instruments. However, some are preferred when it comes to expressive, gestural control. A comprehensive survey on the use frequency of different sensor types in the NIME community was presented by M. T. Marshall, Hartshorn, Wanderley, and Levitin (2009) in 2009. The results, shown in Table 4.1, indicate a preference of force-sensing resistors, accelerometers, camera-based systems and buttons or switches. Native input paradigms of electronic musical instruments, like rotary and linear potentiometers, are left behind. It is likely that the recent progress in camera-based tracking systems has an influence on this development.

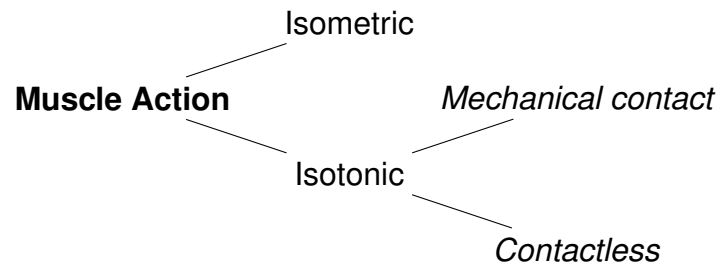


Figure 4.1: Taxonomy of muscle-based interaction, proposed by Bongers (2000).

Table 4.1: Occurrences of different sensor types in NIME instruments (M. T. Marshall, Hartshorn, Wanderley, & Levitin, 2009).

Sensor	Occurrences	Property Sensed
FSR	68	Force
Accelerometer	56	Acceleration
Video Camera	54	
Button/Switch	51	Position (On/Off)
Rotary Potentiometer	31	Rotary Position
Microphone	29	Sound Pressure
Linear Potentiometer	28	Linear Position
Infrared Distance Sensor	27	Linear Position
Linear Position Sensor	23	Linear Position
Bend Sensor	21	Rotary Position

4.2 Expressive Control of Sound Synthesis

Many aspects of electroacoustic music and electronic musical instruments are subject to control. One of the oldest and most prominent applications is probably the control of sound synthesis processes, most notably of melodic instruments. The past 100 years have produced a plethora of novel devices and concepts for controlling electronic musical instruments and especially synthesizers. A comprehensive treatise on this topic is thus beyond the scope of this thesis. Yet, insights into the key issues, the history and related approaches help to better understand the design of the haptic interface developed within this project, introduced in Part 9.

In most musical instruments with mechanical sound production, the sound is generated and controlled through immediate interaction with oscillatory and resonating elements. According to the taxonomy by von Hornbostel and Sachs (1914) this includes idiophones, membranophones, chordophones and aerophones. Depending on the instrument and its principle of sound production, the level of immediacy varies. Interaction is certainly most direct for the human voice, where no technical apparatus is involved at all and oscillation, as well as the properties of the resonant body can be manipulated. For wind and bowed string instruments, the excitation is tangible and still very direct. In harpsichords and pianos, sound production and control are already separated by a mechanical construction and the degrees of freedom are reduced. For organs, this separation is even more obvious. The keyboard can be thus be regarded an interface for interacting with sound production systems.

Instruments with electronic sound production, both analog and digital, do not implement an immanent way of control. They always require an arbitrary interface for controlling the excitation and timbral qualities, which is usually designed separately. For the mere generation of sound, control becomes obsolete. For expressive synthesis, it does not. This is equally a chance and a problem, since the connection between sound generation and control is only based on design decisions. According to Julius Smith, the control of digital musical instruments was still “*in its infancy*” in the early 1990s (J. O. Smith, 1991). And still today, this aspect of digital musical instruments is among the most active research topics. Although many research projects and artistic works have addressed this issue since then - and before - no novel concept could replace the keyboard as the standard interface and most likely there is no such thing as the next universal musical interface. However, at this point in time, the means for adding expressive capabilities to digital synthesis systems are growing, rapidly. The constant increase in computer performance, the recent rise of DIY technology, including sensory equipment and development boards for microprocessors, as well as the rapid evolution of wireless communication capabilities offer seemingly limitless possibilities. Based on this evolution we are likely heading for a scenario with many custom solutions for different use cases and requirements.

Over the years, the NIME community has identified several guidelines in the design of interfaces for sound synthesis. Levitin, McAdams, and Adams (2002) suggest a list of activities and actions which need to be considered when designing musical instruments:

- select a tone from among a ‘menu’ of tones
- start a tone playing
- add a tone to an ongoing tone (in polyphonic instruments)

- stop a tone from playing
- take away a tone from an ongoing set of tones (in polyphonic instruments)
- create vibrato (small, rapidly oscillating shifts in pitch)
- create tremolo (small, rapidly oscillating shifts in volume)
- trill (alternate rapidly between two or more tones)
- glissando (slide to a new pitch, producing continuous pitch change in between)
- select a particular dynamic
- change dynamics (louder or softer)
- select a particular timbre
- change timbre
- create various timbral or spectral effects specific to the instrument (e.g. ‘growling’ on the saxophone; feedback or harmonic shifts on the electric guitar; pedalling on the piano; various embouchure changes)

These considerations are limited to traditional musical ideas, as found in melody instruments for western music and the instrument metaphor is thus rather narrow. More general, Hunt and Wanderley (2002) proclaim four rules for the design of interfaces for expressive digital musical instruments with a focus on sound synthesis:

1. require energy for amplitude
2. two hands are better
3. use complex mappings—changes to one parameter should inflect others
4. control timbre in a non-direct manner

Rather accidentally, the interface designed in the course of this thesis is based on some of these rules, especially the connection between energy and amplitude and the two-hands principle. Malloch et al. (2019) present the MIDWAY project, a *design workbench for interactive music systems*. Within their project, the following four guidelines were added to the existing ones:

5. match integrality and separability of controls
6. consider the speed of interaction when choosing inputs
7. support personal strategies and representations
8. use multiple parallel representations

This second set of guidelines focuses mainly on the mapping strategies of the instrument, which will be discussed in detail in Section 4.4.

4.2.1 Historic Concepts for Expressive Control

Before synthesizers with traditional keyboard had their breakthrough in popular music, electronic musical instruments of the early 20th century already explored different ways of expressive control. Common characteristics of some specimen are continuous control parameters to allow expressive techniques like vibrato and glissando.

The *Theremin* is probably one the most known and popular instruments based on an alternative control concept. Patented in 1928 by Leon Theremin (Nikitin, 2012), the synthesizer is controlled contactless in mid-air through electrostatic capacitive sensing. Two antennas are integrated into the circuits, allowing the independent, continuous control of intensity and pitch through proximity of the hands. The simple synthesis circuitry, combined with this ironically very immediate control principle, results in a very expressive instrument, reminding of the human voice. As a result, the Theremin spawned several renowned performers, as for example Clara Rockmore or Carolina Eyck, playing both adapted pieces from the classic repertoire as well as original compositions.

The *Trautonium*, invented by Friedrich Trautwein in 1930 (Sala, 1950), is a subtractive synthesizer with continuous frequency control. This is realized through a conductive wound string stretched over a metal rail. By pressing the string down to the rail, an electrical contact is closed. The pitch of the resulting sound depends on the position of the contact, spanning a range of over 3.5 octaves. Since the string is continuous, this enables the application of vibrato and glissando, as applied on a violin string. Additionally, the octave can be shifted to enlarge the pitch range. Auxiliary keys above the string provide guidance for identifying the pitches **c**, **g**, **d** and **a** in every octave. A particular feature of the instrument is its synthesis principle, based on a source filter model. A sawtooth oscillation is treated with bandpass filters, resulting in vowel-like timbres. The Trautonium was immediately accepted as a novel instrument in the classical context, promoted especially by composer Paul Hindemith (Brilmayer, 2014).

The *Ondes Martenot*, invented by Maurice Martenot in 1928, is another example for an early electronic instrument with a focus on expressive control. Pitch is controlled with the right hand, using either a conventional keyboard or a continuous ribbon moved through a finger ring. This enables glissandi and vibrato, as in the Theremin or the Trautonium. The left hand controls the intensity with a separate force-sensitive key, allowing a wide range of articulation styles. Different waveforms can be selected for changing the timbre. The sound is further shaped through different loudspeaker or transducer concepts which are part of the instrument. Several composers of the early 20th century, among them Olivier Messiaen and Maurice Ravel, contributed more than 1500 compositions for the Ondes Martenot. (Quartier, Meurisse, Colmars, Frelat, & Vaiedelich, 2015) Messiaen was in particular impressed by the expressive means offered by the independent intensity key.

4.2.2 Control in Classic Synthesizers

Most established devices used for controlling synthesizers are borrowed from their mechanical predecessors and the keyboard still is the predominant control device. Especially the popular classic synthesizers in the history, including for example the Minimoog, the Korg MS-20, the

Roland Jupiter and Juno Models, Oberheim synthesizers and the Yamaha CS 80, are all equipped with keyboards. Indeed, it appears to be the appropriate interface for polyphonic synthesizers, making them wholesome harmony instruments. For monophonic melody instruments, however, this polyphonic concept alone is rather inadequate, since it lacks possibilities of expressive control.

Various attempts have been made to improve the expressive means of keyboard-based interfaces, already in the early stages of analog synthesis. Some of them have established and are now indispensable in electronic keyboards. The Minimoog Model-D was the first instrument to feature the *mod wheel* and the *pitch wheel* (Pinch, 2015), which immediately became the most common augmentations for expressive control in keyboard synthesizers. These devices allow musical articulation and ornamentation and shaped the characteristic play of keyboard virtuosos like Chick Corea or Jan Hammer. The pitch wheel is used for direct control of frequency deviation, whereas the more versatile mod wheel usually controls the depth of an LFO driven modulation. Both can be considered original means for expressivity on electronic musical instruments (Arfib et al., 2005). A wide range of combinations and modifications of the mod wheel and pitch wheel can be found in keyboards of the past decades, including joysticks and touch sensors or versions on *keytars*, synthesizers played like electronic guitars, as for example the Yamaha KX1, the Roland AX-7 and the Roland SH-101.

Velocity-sensitive synthesizer keyboards allow the accentuation through the force of the impact. Similar to the piano, not only the level, but also the timbre of synthesized sounds is affected by the velocity of the stroke. Another well established concept, closely related to velocity, is the *aftertouch* of synthesizer keyboards. Once a key is pressed, the continuous force applied to it is used as a modulation parameter which can be routed to arbitrary synthesis parameters. Early synthesizers with aftertouch are the Multimoog from 1978 and the Yamaha CS80 from 1977 (Jenkins, 2019) or the Korg Sigma from 1979.

Besides these specific inventions, rotary knobs, sliders and faders are the native control elements for electronic musical instruments. In different genres and musical practices, expressive and virtuous techniques have emerged based on these simple input devices. This includes filter sweeps with a potentiometer and cross fades between different sounds, as for example in DJing.

4.2.3 Alternative Control Principles

Keyboards and Multi-Touch Devices

Similar to the early Trautonium or Ondes Martenot, more recent alternative keyboard-inspired concepts for expressive control have been invented. Especially 2D multi-touch interfaces have gained popularity as general purpose musical controllers, due to their distinguished expressive capabilities. A recent survey on this family of devices is presented by Schwarz, Liu, and Bevilacqua (2020). The overview includes some smaller sized touch interfaces like the *Morph* by Sensel, the *Kaoss Pad* by Korg or the Roli *Lightpad*, not necessarily intended for melodic play but for general purposes. Other examples have the dimensions of actual piano keyboards and are specifically designed for controlling pitched instruments.

TouchKeys (McPherson, 2012) augments a conventional MIDI keyboard with capacitive touch

sensors. A 0.8 mm printed circuit board is mounted on top of every key, featuring 25 discrete sensor pads for the white keys and 17 for the black keys. This allows the tracking of the touch position on individual keys, as well as pinch and slide gestures, transmitted via OSC in addition to the MIDI data from the underlying piano. These extended possibilities allow the application of vibrato and detailed articulation styles with the familiar tactile feedback of a piano.

The *Roli Seaboard* is a rubber-keyed piano (Eom et al., 2018; Roli Ltd., 2020), combining a traditional piano layout with a continuous frequency control and extended articulation possibilities. It can hence be played with skills from the conventional keyboard, making it accessible to a larger target group. Additional expression is granted through sensing of applied force and of the touch position on a key. Polyphonic continuous frequency changes are possible by sliding the fingers on an area in front of the keys.

The *Continuum Fingerboard* is a multitouch interface for controlling polyphonic synthesis processes (Haken, Fitz, Tellman, Wolfe, & Christensen, 1997; Haken, Tellman, & Wolfe, 1998; Haken, 2004). Up to ten voices can be played simultaneously, tracked by Hall-effect sensors on a board with the size of a traditional keyboard. The position on the board as well as the pressure applied is sensed for each finger. Control is thus three-dimensional, with the latitude controlling the pitch of each sound, as in a piano, the force controlling the intensity and the depth controlling the timbre. This three-dimensional space, referred to as the *timbre space* is accessible through additive synthesis (Haken, Fitz, & Christensen, 2007).

The *LinnStrument* (Linn, 2020) by Roger Linn is a board with a matrix of either 128 or 200 pads, arranged in eight rows and 16, respectively 25 columns. The pads are force-sensitive, whereby expressive three-dimensional control is enabled. On the board, notes are arranged as on a string instrument to facilitate expressive gestures like glissandi, spanning 5 octaves. Due to the matrix layout with RGB backlight, the LinnStrument can also be used as a step sequencer.

Similar to the LinnStrument, the *Soundplane* (Madronalabs, 2020) by Madrona Labs is a board with a walnut playing surface, detecting the position of touch in two dimensions, as well as the applied force. Ten touch points can be used, simultaneously. It can be configured as a keyboard with 150-notes in a 30 times 5 matrix, allowing position and pressure sensing on each key, or as a continuous touch surface. A control rate of 975 samples per second makes the interface suitable for direct control through pressure.

Instrument Specific

A large variety of interfaces is based on the principles of existing mechanical instruments, either by extending them through sensors or by replicating their basic layout. This is in particular meaningful in combination with the related synthesis algorithms for a full emulation of an instrument but also allows trained musicians to control arbitrary synthesizers with a control principle they have mastered.

Especially wind controllers have landed a greater commercial success, since their basic functionality is easily implemented by sensors and adequate synthesis algorithms are rather easy to implement. Early examples are the *Lyricon Wind Synthesizer Driver* from the early 1970s and the well known *Electronic Woodwind Instrument*, invented by Nyle Steiner in the late 1970s and

early 1980s and later produced by Akai (Paradiso, 1997). Sensing of breath- and lip-pressure, combined with capacitive touch sensors, allow swift expressive play. In the late 1980s, Yamaha introduced the WX-7, the first wind controller with MIDI output to the market. Besides many commercial devices, the research community continued exploring enhanced wind controller concepts, using for example inertial measuring units for motion tracking (Henriques, 2008).

Musical interfaces based on string instruments are technically more challenging than keyboard-inspired approaches. In the 1970s, the hexaphonic magnetic pickup first enabled the proper use of an electric guitar as a control device (Paradiso, 1997). Since every string has an individual pickup, an independent control signal can be generated from all six strings, using tracking of pitch and energy envelope. The resulting MIDI signals can be used to control polyphonic synthesis processes. Today, a considerable amount of guitars is available using this method. Other concepts adopt only the neck without strings as a surface for pitch control, making use of potentiometers to track the left hand position (da Silveira, 2018).

Since bowing techniques contribute a great part to the expressive capabilities of bowed string instruments, violins and other related instruments have been augmented into musical interfaces in different projects. This paragraph lists only a few relevant examples. In the course of the *Hyperinstrument* program at MIT, continuous control parameters were generated from specially built string instruments. Manual input was considered, as well as fully automated control streams and hybrid forms. Starting with the *Hypercello* in 1990, a customized bow allowed to use bow position and wrist angle, as well as finger position and pressure on the board (Machover, 1992). Each of the four strings is picked up, generating four independent loudness and pitch parameters. Control data was sampled with 100 Hz. After conversion to MIDI data, the control parameters were used to drive commercially available samplers and synthesizers. The *Hyperbow* (Young, 2002), is a musical interface, based on the Hypercello. The bow is equipped with sensors for capturing position, acceleration and multidimensional strain, still providing the musician with an instrument of acceptable size and weight. Sensor data is wirelessly transmitted from the bow and processed with sampling rates of $40 \dots 140$ Hz. The *BoSSA* (Trueman & Cook, 2000) is not an interface but a full-fledged musical instrument, combining the violin's physical performance interface with a synthesis algorithm and a speaker array for the emulation of radiation patterns. Its interface consists of a standard violin bow, fitted with pressure and motion sensors, and a single-stringed violin fingerboard with a linear position sensor and four foam-covered force sensing resistors as strings. Without aiming at a violin emulation, synthesis in the BoSSA is, for example, sample-based with bow-strokes playing back pre-recorded sentences.

Gloves and Related Devices

Sensor-equipped gloves for tracking orientation and posture can turn performers hands into highly expressive interfaces for musical interaction. After first attempts were made with early *data gloves* – general purpose or virtual reality gloves, like the *Sayre Glove* from 1977 or *MIT Data Glove* from the 1980s, gloves were specifically designed for digital musical instruments Kestelli (2019). *The Hands*, developed and performed by Michel Waisvisz since the early 1980s for over 20 years, represent an outstanding project in the field of gestural sound control (Torre, Andersen, & Baldé, 2016). Throughout three consecutive versions, the bi-manual mid-air inter-

face was co-developed with dedicated synthesis and processing hardware and software, resulting in a full-fledged musical instrument. Practically a successor, *The Lady's Glove* by Laetitia Sonami went through five different versions, starting in 1991 (Sonami, 2020). Version three features bend sensors for the fingers, a pressure pad on the index finger and distance sensors for tracking the elevation of the hand, later replaced by accelerometers. Sensor signals are processed in STEIM's *Sensorlab*, stemming from the work on *The Hands* and mapped to MAX-MSP for sound generation. The *MusicGlove* (Hayafuchi & Suzuki, 2008) is a wireless glove with acceleration sensors and strain sensors for the wrist and two fingers. A fixed set of gestures, like finger bending, wrist rotation and hand motions, controls audio and video processes in Max/MSP for virtual DJing or conducting. The *Mi.Mu* gloves, a project started by Tom Mitchell and Imogen Heap under the name *SoundGrasp* (T. Mitchell, 2011), is designed for live sampling and looping. Posture and orientation of the hands are captured with bend sensors and inertial measurement units, respectively.

Smaller wearables can be easily integrated into existing setups and workflows of electronic musicians and are thus gaining in popularity. The *Genkli Wave* (Genki Instruments, 2020) is a finger ring with an embedded IMU and wireless transmission, designed for controlling arbitrary musical parameters. It comes with a Eurorack receiver for using it in modular synthesizer setups. Although the *Sonic Swarm Controller* (Davis & Karamanlis, 2007) works similar to commercial devices for hand tracking, it is a custom built interface for a specific spatial music application. A swarm of soundfiles is controlled by a hand-mounted IMU with additional rotary potentiometers and keyboard commands, using Max/MSP. Tilting the hand along different axes influences pitch, window size and grain position in a granular swarm. Vertical position of the hand influences the volume and a filter cutoff.

Handhelds

The advances in sensor technology, wireless communication and DIY electronics allow the integration of diverse sensors in arbitrary hardware configurations. Some resulting devices bear a resemblance to the interface developed within this thesis.

Hoelzl, Han, and de Campo (2019) present the *nUFO* (nontrivial/new flying object), a digital musical instrument with a haptic interface, designed for controlling a sound library of 24 pre-defined sounds. The disc-shaped control device is operated in mid air with two hands and shows technical and conceptual similarities to the interface developed in the context of this thesis, introduced in Part 9. It includes an inertial measurement unit and eight capacitive sensors for measuring the pressure of the fingers when holding the device. A *MetaControl* layer for mapping, referred to in Section 4.4.4, and a *Sound Library* for sound generation complete the instrument.

4.2.4 Partial Performance Synthesis

Partial performance models can be considered assistance systems for the expressive control of musical instruments. Actual performance synthesis or performance rendering refers to the interpretation of musical content through computer systems. A system for performance synthesis can for example be fed with a musical structure, such as a MIDI file or a symbolic score, and

generate control trajectories for driving sound synthesis (Dannenberg & Derenyi, 1998). This facilitated by creating performance models in advance, for example through music information retrieval and machine learning from recorded performances (Widmer & Goebel, 2004). A performance model includes the interpretation style of the musician and the individual instrument, but also depends on the acoustical properties of the recording site and the musical content used for gathering performance data. Besides increasing the expressive capabilities of electronic musical instruments through immediate control, certain aspects of musical performance can be added by the instrument. Such *partial performance models* (De Poli, 2004) let the musician control the majority of expressive directly, adding only few aspects. This is further referred to as *partial performance synthesis* and was already included in early analog synthesis systems.

As introduced in Section 4.2.2, the modulation wheel, or *modwheel*, is used to control low frequency oscillations (LFO) for a semi-automated vibrato. It can be found in early synthesizers, such as the Minimoog. Although the synthesizers offer the control of various parameters, such as modulation frequency, and depth, the musician is offered a previously designed model with included expressive content. The LFO is also considered a category of control by (Verfaillie, Wanderley, & Depalle, 2006). Another example is the semi-automated realization of glissandi in synthesizers, also featured in early synthesizers (Olson & Belar, 1955; Deutsch, 1978), sometimes called portamento. When played in the *portamento mode*, synthesizers automatically shape the frequency slope between two successive notes of different pitch. Depending on the synthesizer, performers can choose among different slope types, such as linear, exponential or tangential, which are pre-defined by the instrument designers. The basic nature of the slopes stems from analyses of recorded glissandi on existing instruments, considering their pitch trajectories.

Most of the more advanced systems for expressive sound synthesis introduced in Section 3.4 feature partial performance synthesis. Especially concatenative systems, which use full segments with expressive musical content rely on this approach (Maestre et al., 2009). Gestures are hence determined by the corpus. Another example is the performance sampling approach presented by Bonada and Serra (2007), which can be driven by a musical score, as well as by real-time expressive input from a musician. The Reconstructive Phrase Modeling system presented by (Lindemann, 2010) and introduced in Section 3.4.1 (p. 42) allows different levels of expressiveness in the control stream. More detail is contained in the control data when using an expressive interface, as for example a wind controller, then when using a MIDI keyboard or a simple piano roll. In the latter case, the system can add expressive musical content, based on the model created during analysis.

Immediate control and partial performance synthesis are by no means exclusive. It is most likely a combination of interfaces and machine learning approaches which will make modern synthesizers more expressive. Within the synthesis system proposed in this thesis, certain expressive aspects are extracted from recorded material, whereas other underlie direct control.

4.3 Control and Spatialization

4.3.1 Historical Viewpoints

Although musicologists agree that the spatial qualities have been considered by composers and performers alike throughout the history of music, the advent electroacoustic technologies introduced entirely novel means of working with sound in space, dynamically. Methods for controlling the spatial properties of music are thus deeply rooted in electroacoustic music and have already been conceived in the early years.

In the 1950s, Pierre Schaeffer developed the *Pupitre d'Espace*, a device based on magnetic induction for the diffusion of *Musique Concrète* on quadraphonic loudspeaker systems (Pysiewicz & Weinzierl, 2017). Four receiver rings are placed around the performer in 50 cm diameter, arranged according to the loudspeaker positions. A performer, also referred to as *operator-conductor*, holds a coil with alternating current of 1 V at a frequency of 5 kHz (Battier, 2015). The closer the coil is held to a ring, the louder the signal on the corresponding loudspeaker. Through broad gestures, the sound could thus be moved within the reproduction system. Although there has been a considerable progress in sensor technology and signal processing, this expressive way of mid-air control is as up-to-date today as it has been when it was conceived.

Just as *Musique Concrète*, Cologne-based *Elektronische Musik* also worked with space as a shaping parameter since its beginnings. Karlheinz Stockhausen invented the rotation table for creating movements of sound on loudspeaker systems. This device is equipped with a loudspeaker in the center and four stationary microphones placed in a square around the table. When spinning the table, the radiation direction of the speaker is changed and the microphones receive a different amount of the sound. Four tracks are recorded, one for each microphone. If they are synchronously played back on a quadraphonic setup, the rotation is reproduced. This rather low-tech mechanical procedure even contains doppler shifts, resulting in convincing spatial gestures, as heard in Stockhausen's *Kontakte*.

While the rotation table was only used in studio production, other examples from the history of German electroacoustic music were designed for live spatialization. The *Tonmühle*, developed by Manfred Krause at Technical University Berlin, was inspired by Stockhausen's rotation table but based on an electronic principle (Gertich, Gerlach, & Föllmer, 1996). By turning a crank in circular motion, a slip ring potentiometer distributed the monophonic sound source to ten loudspeakers, depending on the angle. Due to several drawbacks, the device was only used in experiments at TU Berlin.

The Philips pavilion at the 1970 world fair in Osaka was a milestone project for spatial electroacoustic music. 50 groups with each 13 loudspeakers were mounted in a spherical auditorium, playing pieces by Stockhausen, Bernd Alois Zimmermann, Boris Blacher, Beethoven and others. A spherical control device for live spatialization in the auditorium was developed and implemented by students at Technical University Berlin (Gertich et al., 1996). 50 buttons were arranged on the surface of the handy device. The level of each speaker group could be controlled by pressing the belonging button, allowing swiping gestures. Since the operation was not trivial, the control device was only used for Blacher's piece *Große Kugelkomposition*.

At the SWR studio, Hans Peter Haller and Peter Lawo invented the *Halaphon* (Haller, 1995, 1995), a spatialization system for elaborate mixed music performances. The device allowed the programming of spatial trajectories and patterns on arbitrary loudspeaker systems through panning, most notably used for the works of Luigi Nono, for instance the ambitious spatial realization of *Prometeo*, premiered in 1984. Live input of instruments could be used to control the system, in order to trigger sound movements (Brech & von Coler, 2014, 2015).

4.3.2 Parameters of Spatial Control

Spatial control is in this thesis treated from a point source panning approach, as introduced in Section 3.5.3. At the center is thus the manipulation of relevant properties of virtual sound sources. Kendall and Ardila (2008) point out a lack of terminology and theoretical constructs for spatial attributes in the field of electroacoustic music. However, such a framework is necessary for identifying and implementing relevant parameters of control for spatial sound phenomena. Research on the perception of real and virtual acoustic scenes, as well as musicologists' views on spatial electroacoustic music deliver a coherent set of attributes and parameters.

Lindau et al. (2014) propose a vocabulary for the qualitative assessment of real, imagined and simulated acoustic scenes, based on a focus group method with 21 experts. The full set features 48 verbal descriptors, organized in the 8 categories *timbre*, *tonalness*, *geometry*, *room*, *time behavior*, *dynamics*, *artifacts* and *general impressions*. Many of these attributes would be meaningful for a thorough inspection of the instrument presented in this thesis. Yet when focusing on the control of spatial sound synthesis, we focus on the category *geometry*, which consists of 10 Perceptual qualities listed in Table 4.2. The descriptors *front-back position* and *externalization* relate primarily to the evaluation of binaural and headphone-related systems and can be ignored in this context. This leaves three positional parameters (*horizontal direction*, *vertical direction*, *distance*) and three parameters of spatial extent (*depth*, *width*, *height*) for describing the virtual sound source. The two additional qualities *localizability* and *spatial disintegration* are of special interest in the scope of this thesis. As explained in Section 3.5.3, spatial sound synthesis allows to treat them as parameters of musical expression.

The perceptually relevant parameters presented by Lindau et al. (2014) are also found in the discourse on electroacoustic music and spatialization. M. Marshall, Malloch, and Wanderley (2007) define three categories of parameters for the control of sound spatialization:

1. sound source:
 - (a) position and orientation
 - (b) characteristics
2. environmental and room model parameters

In this taxonomy, position and orientation includes the parameters position in Cartesian coordinates (x,y,z), respectively in the more common spherical coordinates *azimuth*, *elevation* and

Table 4.2: Perceptual qualities in the category *geometry* of the SAQI (Lindau, 2015).

Quality	Description	Scale
Horizontal direction	Direction of a sound source in the horizontal plane.	shifted anticlockwise -shifted clockwise (up to 180 degrees)
Vertical direction	Direction of a sound source in the vertical plane.	shifted up - shifted down (up to 180 degrees)
Front-back position	<i>Confusion between positions in front or back.</i>	<i>dichotomous scale: not confused / confused</i>
Distance	Perceived distance of a sound source.	closer - more distant
Depth	Perceived extent of a sound source in radial direction.	less deep – deeper
Width	Perceived extent of a sound source in horizontal direction.	less wide – wider
Height	Perceived extent of a sound source in vertical direction.	less high - higher
Externalization	<i>perceived within or outside the head</i>	<i>more internalized – more externalized</i>
Localizability	If localizability is high, a sound source is clearly delimited.	more difficult – easier
Spatial disintegration	unwanted spatial separation.	more coherent – more disjointed

distance. The characteristics are composed of *size*, *directivity*, and *presence / distance*, as well as the timbral characteristics *brilliance / warmth*. Directivity and timbral features are not considered parameters of spatial control in the scope of this thesis. Size is in this model not further divided into single dimensions. The third group, the room parameters, include the items *size*, *presence*, *early reflections*, *reverberation*, *reverb. cutoff freq.*, *Doppler effect*, *equalization/filtering*, *air absorption*, *distance decay*, *mic/sink/speaker position*, *mic/sink directivity* and *heaviness/liveness*. Although relevant for spatial music in general, they are not considered in the control model for spatial sound synthesis. Many of them can be influenced by the rendering algorithm, inherently.

Similar to the above introduced SAQI model, Kendall (2010) proposes a parameter structure with *dimensional attributes* and *immersive attributes* as main categories. Within this model, the dimensional attributes are comprised of *direction*, *distance* and *extent*. The latter is again divided into *depth*, *width* and *height*. The immersive attributes include the features *presence* and *envelopment*. These considerations refer to experiments by Zacharov and Koivuniemi (2001) and Berg and Rumsey (2003), who present methods for evaluating different spatial reproduction techniques and different microphone techniques. Spatial sound synthesis allows to change the presence and envelopment, dynamically.

Peters (2011) gathered 47 responses from composers in an open-ended form on the compositional means for creating spatialized sound experiences. The resulting list is divided into three categories *distribution and position*, *movements* and *others*, as listed in Table 4.3. Central aspect in the responses are methods for “*moving sound sources and their distribution in space*” (Peters, 2011, p.33). *Distribution and position*, includes relevant parameters for the control of spatial sound synthesis. Especially *source distance and depth*, *spatial organization*, *diffuse sounds* and the *size of the spatial image* are congruent with the above introduced parameters and relevant for the system developed in this thesis.

Table 4.3: Composers' responses on spatial compositional means (Peters, 2011, p.34).

Distribution and position	Distance and depth
	Algorithmically generated distributions
	Spatial organization according to timbre, texture, and musical function
	Spatial granularization
	Stereo tracks as source material
	Diffuse sounds (multi-speaker distribution)
	Mono sound reproduced from one or two adjacent loudspeakers
	Size of the spatial image
	Planes, subspaces and hierarchical sound layers
	Front (stage) is the focal point
Movements	Slight
	Contrast
	Many movements with a small number of sounds
	Strength of movements according to musical function: melody moves more than other sounds
	Prepared trajectories
	Instrumentalists move during composition
	Live performers moving loudspeakers
Others	Changing loudness and dynamics
	Spectral filtering
	Reverberation
	Simulating room acoustics
	Surreal spatial impressions

4.3.3 Devices for Spatial Control

Although the development of control interfaces for spatialization and related tasks is as old as the spatialization itself, rather conservative approaches, such as faders, mixing desks, cues, envelopes, and programmed trajectories remained the standard in many applications, as for example in the composition and diffusion of acousmatic music (M. Marshall et al., 2007). In the context of *Timbre Spatialization*, Normandeau (2009) pointed out the need of 3D audio controllers in 2009. Control of spatial sound is still a challenging task and hence an active field of research (Garcia et al., 2017). Depending on the kind of interaction, these controllers may have a very different focus. M. Marshall et al. (2007) distinguish between the *spatial performer*, who has direct influence on sound objects, the *instrumental performer*, controlling spatial attributes indirectly and the *spatial conductor*, controlling whole auditory scenes, as for example in sound diffusion. For spatial sound synthesis these roles merge into one.

Compared to sound synthesis which is also applicable in a popular music context, spatial sound techniques largely remain in the field of experimental music. Consequently, less interfaces for controlling spacial parameters of music are commercially available. Often they are based on universal input devices or developed as prototypes for single applications. Johnson, Norris, and Kapur (2014) present a review on interfaces for spatial control and identify three categories, namely *existing tools*, often lend from the gaming community, *multi-touch interfaces* and *entirely new*

interfaces from the NIME community. Perez-Lopez (2015) lists a selection of *spatialization instruments* and compares their spatialization parameters, as well as their features as digital musical instruments. The following section lists only a few relevant examples, arranging them in four categories.

Multi-Touch and Tabletops

Multi-touch interfaces, based on touch screens or other touch-sensitive surfaces, offer quasi native means for controlling object-oriented spacial scenes in two dimensions. Performers can directly control the coordinates of virtual sound sources or sound objects in the horizontal plane. A diffusion interface based on the *BrickTable* multi-touch interface for musical performance is presented by Johnson and Kapur (2013). The system offers quadraphonic panning, stereo pairing and VBAP as rendering techniques for embedded sound files or live input. In the most basic mode, users can place a visual representation of an audio object within a representation of a loudspeaker setup.

Tabletops can be played by touch but also by placing and moving physical objects on the surface. In this way, sound objects can be linked to actual tangible objects for two-dimensional spatialization. (Sasamoto, Cohen, & Villegas, 2013) present a table-top interface for controlling VBAP on a ring of eight loudspeakers. Multiple users can place four square beverage coasters on the circular surface to determine the position of virtual sound sources with different waveforms from Pure Data patches.

Gloves

Data gloves are especially suited for interacting with spacial sound, since they offer means for actually grabbing virtual sound sources and manipulating them with gestures. The system for gestural sound spatialization by M. Marshall et al. (2007) uses different data gloves as an option of interfacing. The wearables use bend sensors on the fingers and a 6 degree-of-freedom position tracker. They are not only used for controlling the position of virtual sound sources but also other spatial aspects, such as room parameters and parameters of sound synthesis. A similar glove for tracking nine parameters of position data as well as the hand posture with five sensors is presented by Schacher (2007) A transformation matrix maps the interface data to spatial sound movement, using Max/MSP and Pure Data.

Motion Capturing

Different available systems allow motion capturing of the whole body or single limbs, also referred to as *Natural User Interfaces* - NUI (Grani et al., 2015). Usually not explicitly designed for the control of spatial sound, they are specifically designed as gestural interfaces and have therefore been applied in many music-related projects. Just as touch displays appear to be the native interface for two-dimensional spatialization, motion capturing acts as a natural control paradigm for sound in three dimensions. Applications are often embedded into virtual or augmented reality concepts with gaming character.

Motion capturing systems for single limbs can be based on small, sensor-equipped wearables or handheld devices. M. Marshall et al. (2006) use the *Polhemus Liberty* electromagnetic tracker for gestural control of sound spatialization. Still wired, the sensors track 3D position and orientation at a sampling rate of 240 Hz. A *ViMiC* (Virtual Microphone Control) spatialization system is implemented for a circular eight channel loudspeaker setup in Pure Data and Max/MSP. The authors propose a *Gesture Description Interchange Format* (GDIF), structuring OSC messages by their level of abstraction, for easier mapping onto spatial parameters.

The *Wiimote* is a gaming controller by the Nintendo company, released in 2006 and frequently used in experimental music projects. The *Grainstick* project combines WFS and corpus-based concatenative synthesis with a gestural interface (Leslie, Zamborlin, Jodlowski, & Schnell, 2010). In a collaborative scenario, two persons can span a virtual rainstick between two Wiimote controllers. The tilt of this virtual tube can be controlled by the relative height of the controllers, resulting in silence for equal level of both controllers and a dense avalanche of grains for a steep tilt.

The *Microsoft Kinect*, an infrared-based motion capturing camera, was released in 2010 as an input device for the *Xbox* gaming console. Since it was significantly cheaper than other devices with similar functionality and easily integrated into music programming languages, it instantly became popular and was used in various musical applications. The *Leap Motion*, released in 2013, works with the same principle as the Kinect, yet for tracking only the hands of a user. It is placed on a table in front of the user and can thus be integrated seamlessly into computer music performances.

Ratcliffe (2014) consider both the Leap Motion and the Kinect in a project focusing on mixing of conventional music in a stereo image. Sound sources can be placed and moved on a virtual stage with hand gestures, with traditional panning and depth as parameters. Sensor and audio processing are realized in Max/MSP and Ableton Live.

Specter (Manaris & Stoudenmier, 2015) uses Leap Motion, Kinect and smartphones, combined with music information retrieval (MIR), as an environment for sound spatialization. The implementation is built around *JythonMusic* for MIR, audio file playback and VPAB or Ambisonics rendering. *Iannix* is used for modeling sound spatialization trajectories.

The wireless *Myo* armband combines an inertial measurement unit (IMU) with electromyography (EMG) sensors for universal gestural control. The *SoundMorpheus* system (Benson, Manaris, Stoudenmier, & Ward, 2016) connects multiple of these devices to a hub, driving MID instruments and Pure Data for sound synthesis and Ambisonics rendering. Orientation and movement of the arm as well as gestures, including *fist*, *spread fingers*, *waving* and *tapping* are tracked for multiple users. Applications include a duet, with one person playing a conventional instrument, fed into the system, and a second person, manipulating and spatializing the sound with the armband an an array of eight loudspeakers. After grabbing the sound of the instrument, its angle of incidence can be changed. *gSpat* (Di Donato & Bullock, 2015) is system similar to the SoundMorpheus, based on the Myo and Pure Data for stereo panning.

Marentakis, Peters, and McAdams (2007) present augmented turntables for spatialization on an eight channel circular speaker setup. A camera is used for tracking hand gestures when

interacting with conventional vinyl, supported by a colored marker worn on the hand. Thus, original sound can be used without the need of time-code vinyl. The angular displacement of the DJ's hand is directly used as an input parameter. With an adjustable scaling it can be used to control the angle of incidence for the sound played back from the vinyl. A Max/MSP based VBAP spatialization is used in the prototype for a ring of eight loudspeakers.

An early system for using Wave Field Synthesis with multi-viewer stereo displays is presented by Springer et al. (2006). Four loudspeaker panels by IOSONO were used for sound field rendering. Viewer position and orientation were tracked with an *ARTrack2* optical tracking system. A hand held trackball allowed interaction with a virtual environment, used for small interactive installations, with sample-based sound synthesis. Naef and Collicott (2006) use a similar system for spatial sound performance. It uses a stereoscopic projection and amplitude panning with 14 loudspeakers, controlled by head- and hand tracking and the *Immersion CyberTouch* gloves with bend sensors for all fingers and haptic feedback. Virtual sound objects can be grabbed and moved and changed in volume through hand rotation.

Fohl and Nogalski (2013) use mid air hand gestures to control positions of virtual sound sources on a Wave Field Synthesis system. The WFS system uses the *WONDER* software (M. A. Baalman & Plewe, 2004) for rendering on 208 channels. An *A.R.T* infrared camera based tracking system is used with hand-mounted 3D markers. This allows the tracking of the hand's position, as well as the orientation in the three angles *pitch*, *roll* and *yaw*. Based on these control parameters, a basic set of gestures is defined, featuring *selection*, *deselection*, *circular movement* and *radial movement*.

The *BoomRoom* is a system for *touching, grabbing and manipulating sounds in mid-air* (Müller, Geier, Dicke, & Spors, 2014). A small room with a diameter of 3 m is equipped with a 56 speaker WFS system behind curtains. An *OptiTrack* system with 16 cameras tracks markers on the user's head and hands, as well as on the objects in the room. WFS rendering is realized in the *SoundScape Renderer* (SSR), fed with sound synthesis from Pure Data, connected via the JACK Audio Connection Kit. Müller et al. (2014) evaluated the accuracy of localization to be 11.5 cm.

Alternative Interfaces

The increasing accessibility of sensor technology, data communication and microcontrollers has also boosted the development of alternative interfaces for spatial control. These are often part of specific projects and thus only used by small groups or single persons.

The *Sound Flinger* is a square-shaped table top device, reminding of Sockhausen's rotation table. It is equipped with four motorized faders, one on each side, designed to "*throw sounds around a physical space*" (Carlson, Marschner, & McCurry, 2011). Using force-feedback objects in Pure Data, spatialization of two virtual *sound objects* is possible in a physical modeling way.

4.4 Mapping in Digital Musical Instruments

4.4.1 The DMI Model

As outlined in the preceding considerations, both sound synthesis algorithms and means for controlling them have evolved rapidly over the past decades, resulting in a plethora of possible combinations. Since interface and synthesis are not inherently coupled in digital musical instruments, the connection between these basic components can be freely designed. This aspect, usually referred to as *mapping*, is a crucial step in their design, offering unique possibilities but also raising novel problems. Especially the expressive capabilities of an instrument depend on the nature of the mapping to a great extent. Consequently, mapping gained more attention in the past years and is now an active area of interdisciplinary research. The mapping problem is universal to human computer interaction (HCI) in general, where modular approaches emerged to ease the design of interactive systems (Anson, 1982), leading to *user interface toolkits* and *user interface management systems* for making the many design decisions easier (Mackinlay et al., 1990). Although initially mostly concerned with interaction for visual representation, auditory displays and sound synthesis were also considered in early HCI research (Baecker, 1980). HCI research paved the way for more specific fields like musical interfaces by developing taxonomies of input devices (Buxton, 1983) and frameworks for evaluating them. In the theory of digital musical instruments (DMI), the *Interface-Mapping-Synthesis* model, as shown in Figure 4.2, has become the standard (Rovan, Wanderley, Dubnov, & Depalle, 1997; Hunt & Wanderley, 2002; Dahlstedt, 2009). Often this model is expanded by the performer, sometimes by the audience, in order to include the feedback path from sound synthesis to the involved persons. For the scope of this thesis, this is not considered relevant.

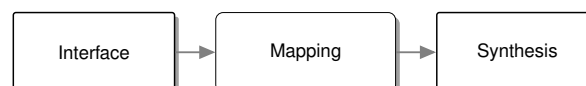


Figure 4.2: General digital musical instrument model, featuring interface, mapping and sound synthesis.

4.4.2 Fusion-Mapping-Fission Model

In systems for musical interaction, a multitude of sensors is used to control a large number of synthesis or processing parameters. Control and synthesis are commonly regarded multidimensional spaces. Instead of connecting sensors with parameters in simple one-to-one connections, the control of digital musical instruments needs to be carried out in a many-to-many fashion. Rován et al. (1997) distinguish between a one-to-one mapping, a convergent mapping (one-to-many mapping) and a divergent mapping (many-to-one mapping). The authors consider the latter to be the *most musically expressive* and thus in general preferable. But in a holistic view of a digital musical instrument, the three possibilities are not exclusive but occur in a system at different stages. Bevilacqua, Müller, and Schnell (2005) refer to the same principle, when they illus-

trate how *many-to-many* mappings result from the combination of *many-to-one* and *one-to-many* mappings.

Based on the interface-mapping-synthesis-model of digital musical instruments in Figure 4.2, a more detailed segmentation is proposed in the course of this thesis. In the *Fusion-Mapping-Fission* model, shown in Figure 4.3, control parameters are generated from multiple sensors in the fusion stage. For the interface developed in the scope of this thesis, explained in detail in Chapter 9, the three parameters of spatial orientation are derived from nine individual sensors of the inertial measurement unit (IMU). This many-to-one fusion can be observed for many sensor systems introduced in the following section. The actual mapping happens between these control parameters and the metaparameters of the sound synthesis system. For example, the two most common metaparameters for melodic musical instruments, also used in the synthesis system introduced in Chapter 8, are pitch and intensity. Finally, the fission block connects the metaparameters with the actual synthesis parameters of the algorithm. Since such systems usually have a more synthesis parameters than control parameters, this is a one-to-many process. According to Rovin et al. (1997), the mapping stage is part of the algorithm in physical modeling synthesis and a one-to-one control is usually sufficient. However, in the case of spectral modeling, as implemented in this thesis, the control of over 200 partial and noise parameters is realized through a few control parameters. In this context, mapping plays a central role, which is inherently embedded in the synthesis system.

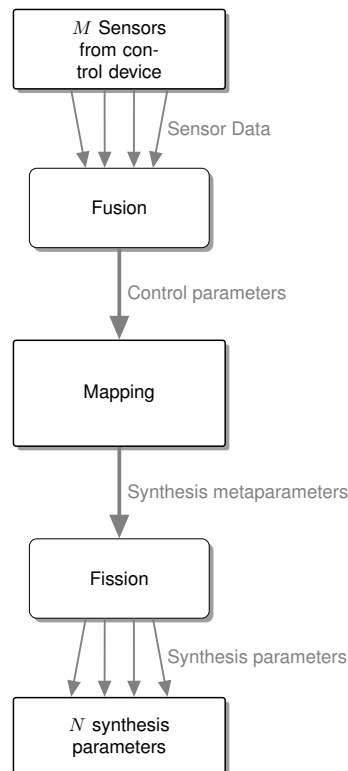


Figure 4.3: Mapping sensors to synthesis, using the *Fusion-Mapping-Fission* model.

Using clearly defined interfaces for the output of the fusion stage, respectively the input of the fission stage, the *Fusion-Mapping-Fission* model allows the quick exchange of different mapping

layers. Since this is a major advantage in the design of musical instruments, similar approaches can be found in this field.

4.4.3 Fingering Systems

Fingering systems come into play whenever combinations of keys, valves or other mechanisms are used for controlling parameters of musical instruments. For digital musical instruments they represent a fusion stage, mapping multiple sensors to the single synthesis parameter pitch. In the context of this thesis, the selection of different pitches through binary combinations, more precisely through pressed/unpressed combinations, is of special interest. This principle is especially meaningful in monophonic applications and can be found in woodwind and brass instruments, for example. For these acoustical instruments, the mapping between binary combinations and tonal resource is, to a greater or lesser extent, predetermined through physical conditions. Key-covered holes in clarinets are an example for working around these limitations with mechanical solutions. The arrangement of valves in brass instruments is also largely based on design choices. Since an inherent linkage between sound generation and control does not exist in electronic musical instruments, fingering systems can be designed arbitrarily. Favorable solutions require minimal changes in the fingering for frequent pitch sequences. The repertoire is thus to be considered.

Beauregard (1991) propose the use of a Gray code for the binary mapping of four bits to one octave. The cost of switching between neighboring positions, respectively pitches in a scale is again considered a measure for the usability of fingering systems in musical applications. Using this measure, the also mentioned *binary* mapping, as used in binary counting, performs worse than the Gray code. The second hand assigns the octave, also based on a Gray code mapping.

Rovan et al. (1997) use a *Yamaha WX7* wind controller to control the timbre of a clarinet sound. Beforehand, the instrument is sampled at three different pitches (F3, F4, F5) and three dynamic levels (pp, mf, ff). Four all nine resulting combinations, an additive model of the sound is calculated. Using an additive synthesis engine, timbres with arbitrary pitch and dynamic in the range of the sampled space can be synthesized through interpolation of the models. Although only briefly introduced in the paper, this synthesis method shows significant resemblance to the approach developed within this thesis, especially to the interpolation procedure introduced in Section 8.1.1. Four mapping schemes are implemented, including a one-to-one example and three different convergent mappings, but not further evaluated.

For controlling expressive voice synthesis, D’Alessandro and Dutoit (2007) present a control device with eight force sensing resistors for selecting the note, completed with a pen-based tablet. The interface aims at controlling pitch and intensity of melodic instruments. Different mappings are explored, as for example controlling pitch, intensity and timbre with a stylus, which did not allow a satisfying amount of expressive gestures. Another mode uses the FSRs to increase or decrease the pitch selected with the pen, resulting in *string-like* fingering techniques.

Hartvigsen (2014) investigates the problem of mapping 7 keys to a possible tone range of twelve octaves, not regarding any other aspects, as for example excitation. Mappings are compared by their ease of use in playing common sequences in classical western music, using a cost function for moving from one pitch to another. The concept of partitionability is introduced in this investigation. A fingering system is partitionable if it operates in a switching mode (M. Baalman,

Bovermann, de Campo, & Negrao, 2014), wherein keys act as shifting keys, to select the octave, for instance. Different fingering systems are favoured for specific scales, confirming that the optimal design is influenced by the content to be played.

4.4.4 User-Defined Mapping

Involving users, respectively musicians, in the mapping procedure of musical instruments can be part of user-driven design, as well as a concept embedded in the final instrument. In the first case, user-defined mapping is used in an iterative design process in final stages of instrument design, for exploring preferred mappings. When the mapping process remains part of the instrument and can be changed by performers, it becomes a co-adaptive system. These systems demand the musician to adapt to their properties, but also allow the appropriation by manipulation of their configuration (Mackay, 2000; Malloch et al., 2019). This section focuses on this aspect of user-defined mapping.

In music programming environments, the boundaries between instrument maker and musician are fluid, since they are instrument, as well as a platform for building instruments. Mapping between control data and sound production or processing is therefore a key element in working with these environments. Many of the examples presented below are thus based on music programming environments, such as Pure Data, Max/MSP, SuperCollider or CSound. Graphical programming languages like Max/MSP and Pure Data (Puckette, 1997) were actually born as patching and mapping environments. Their common ancestor, the *Patcher* (Puckette, 1988), was a graphical environment for connecting MAX real-time processes (Baisnée, Barriere, Koechlin, & Rowe, 1986; Favreau et al., 1986) and for controlling MIDI instruments.

Direct User-Defined Mappings

When integrating mapping possibilities into an instrument, users might need to decide whether they want to configure their instrument or perform. Hunt and Kirk (2000) thus differentiate between the *performance mode* and the editing mode for multiparametric interfaces. The latter allows the changing of system parameters, such as mapping, whereas the performance mode resembles a fixed deterministic system. Instruments with direct user-defined mappings usually show a representation of control and synthesis parameters, allowing to establish and modify connections.

The *Metasurface* (Bencina, 2005) is an interface for the real-time exploration of two-to-many mappings. Users can arrange an arbitrary number of parameter presets on a GUI surface in *AudioMulch*. The surface can then be used for performance, blending between the contained presets through natural neighbour interpolation.

Mapping between control and synthesis parameters is often conceptualized in a matrix paradigm. This matrix can be considered a continuation of the patch bays found in vintage analog and modern synthesizers. In the *modmatrix* Brandtsegg, Saue, and Johansen (2011) every input parameter can be mapped to any synthesis parameter by setting the relevant scaling coefficient. A granular synthesis algorithm with effects, programmed in *CSound* and offering more than 200 parameters.

Multiple settings of the mapping matrix, referred to as *mapping tables*, can be defined, stored and used during performance. A *morphing slider* allows the dynamic interpolation between different mapping tables for additional expressive capabilities.

Mapping environments are not only important for expressive musical instruments but are also helpful for integrating diverse available input devices into computer music contexts. Tahiroğlu (2011) present Pure Data abstractions for using sensors of a smart phone (Maemo Nokia N900) in a multi-user scenario. Touch position, accelerometer, ambient light sensor and other sensors are used to control sound synthesis in a one-to-many fashion. The *6to6Mappr* (Gelineck & Böttcher, 2012) is designed for technological novices, allowing to map control data from MIDI and OSC streams to sound synthesis processes in Max/MSP. Several guidelines ensure the ease of use, among the the restriction to six input and six output parameters, the demand for open source and accessibility but also the ability to output sound with just three mouse clicks. Webcams, MacBook orientation sensors, the Microsoft Kinect, iPhone, Wiimote and Arduino are chosen as input controllers, due to their availability. Sound processing includes audio effects like filters and reverb, granular synthesis, subtractive synthesis and others. Mapping between control and sound processing is possible with dropdown menus and sliders.

Libmapper (Malloch, Sinclair, & Wanderley, 2013) is a software library and protocol designed to create mappings between sensor outputs and synthesizer inputs. Mapper objects are provided for MAX/MSP and Pure Data and bindings are included for Python, Java and SuperCollider. When freely mapping diverse control parameters to a multitude of synthesis parameters, different ranges of sensor data and musical parameters need to be matched in range. *Translation*, *normalization* and *representation standards* are methods for achieving a range matching. MIDI represents a representation standard, defining value ranges for all participants by a strict protocol. In the case of normalization, all data streams are kept within a defined common range, in order to allow free connectivity. Based on the Open Sound Control (OSC) protocol, *Libmapper* data streams and nodes carry information on their range and measuring unit, in order to translate them accordingly.

Influx (De Campo, 2014) is a software library for interacting with processes in computer music contexts. Under the statement *Lose Control Gain Influence* (LCGI), the traditional idea of a deterministic, fully controllable mapping is dissolved. The resulting *MetaControl* paradigm is considered an artistic strategy, rather than a design approach. A similar idea has been proposed by Chadabe (2002) as *interactive instruments*, where the concept of mapping is questioned in general. Key principles for the LCGI approach are *non-triviality* and *entanglement*. Basically, systems are non-trivial if they are non-linear and/or time-invariant. Entanglement refers to co-dependencies among control parameters, realized through a mapping matrix. This matrix connects any number of control parameters with any number of synthesis parameters with individual weights. Snapshots allow the capturing of favourite mappings. The submodules *InfluxMix* and *InfluxSpread* work in a fusion-mapping-fission way, also allowing multi-player mappings. An example of usage is the mapping of a two-dimensional GUI slider to three synthesis processes with six to 16 synthesis parameters.

The *Modality Toolkit* is a mapping library in SuperCollider, intended for quickly mapping modal interfaces to a large variety of sound synthesis and manipulation processes (M. Baalman et al., 2014). Modal interfaces allow the control of many parameters with few input options through

combination. A common example is the shift key of a computer keyboard, opening up a second layer of keys. In several meetings from 2010 to 2014 an interdisciplinary team worked on the *Modality* project to identify common needs and goals. Targets are the inclusion of commercially available interfaces, bidirectional communication with controllers, control stream processing, use of graphical user interfaces and mapping to sound synthesis processes. Mapping in the *Modality Toolkit* is based on transformer modules, matching data from sources or input devices to destinations HID, MIDI and OSC devices can be integrated by defining a device description file. Different modules are part of the library, some based on Functional Reactive Programming and on the above introduced *Influx*. The system also introduces the remap mode. When activated, the performer can swap the assignment of two control elements in a simple one-to-one mapping.

Since the Myo armband is a popular interface in experimental music, a variety of projects created tools for mapping sensor data of the device to sound synthesis and processing software. The *Myo Mapper* (Di Donato, Bullock, & Tanaka, 2018) is a cross platform application, using feature extraction for a better pose or gesture classification.

Brown, Nash, and Mitchell (2018) investigate mapping strategies for musicians using a pair of data gloves. The gloves track the orientation and posture of the hand through bend sensors and inertial measurement units. A dedicated software, the *Glover* allows to connect sensor data with outputs to sound synthesis processes through virtual patch cords, as in known visual programming environments. Two experienced performers and three novice composers one hour to create their own mappings with this tool for controlling a preset of the *Simpler* synthesizer in Ableton Live. Results indicate differences in the mapping strategies for these groups. Experienced users took more time for adapting their mapping throughout the session, whereas novices settled for one solution early which they then tried to master. Overall, resulting mappings were bimanual, using one hand for the pitch control and the other for expressive gestures.

Machine Learning Methods

Since the early 1990s machine learning principles have been applied for mapping gestural control to sound synthesis systems (Fiebrink, Trueman, & Cook, 2009). Initially, models were trained during the design of the instrument and then fixed.

Neural networks, for example, have been trained to connect a set of control parameters with the parameters of sound synthesis systems in the MAX programming environment by Lee, Freed, and Wessel (1991). The *MnM Toolbox* (Bevilacqua et al., 2005) represents a mapping approach based on matrix operations in Max/MSP.

Other approaches have emerged which allow users the training and configuration of machine learning based mappings during practice or performance. The *Wekinator* (Fiebrink et al., 2009) is a metainstrument, enabling users to interactively train machine learning algorithms for mapping controllers to sound synthesis engines. A *Chuck*-based part of the *Wekinator* accepts sensor data but also extracts audio signal descriptors from an input as control signals. Connected via OSC, the *Weka* library is used in java for machine learning algorithms, which do not incorporate any time domain information. Sound synthesis can be implemented either in *Chuck* or externally, through OSC controlled instruments. With these components, users can now train a model by assigning settings of input parameters with sound synthesis parameters in a few minutes. After

training, the resulting mappings can be used in the performance mode.

Verfaillie et al. (2006) present a study on the mapping between input features and adaptive digital audio effect controls. In a *features combination* layer, multiple streams of sound and gesture data are combined and later processed in a *control signal conditioning* layer. The system is used for investigating mapping strategies between gestural control and adaptive digital audio effects. One case example features a dancer, controlling parameters of spatialized granular synthesis. Control is organized in different presets, including direct gestural control over the source position and extent and the control of properties of automated processes.

Sound Tracing

A recurring approach in the research on mapping for music applications is to let participants perform gestures to pre-defined sounds. Machine learning algorithms can then record these gestures and connect them to the resulting sound. This procedure is also known as *mapping-by-demonstration* or *sound tracing* (Tanaka, Di Donato, Zbyszynski, & Roks, 2019).

Caramiaux, Bevilacqua, and Schnell (2009) propose an approach for investigating the relationship between gesture and sound, based on canonical correlation analysis (CCA). 20 participants were presented a variety of sound recordings, consisting of environmental sounds and musical excerpts. They had to perform a gesture matching that sound, with a small hand-held device for camera-based motion capturing. Three-dimensional trajectories for position, velocity and acceleration are recorded and combined with the instantaneous audio features *loudness* and *sharpness* by means of a multivariate analysis method. For sound textures the results show a connection between the loudness feature and the trajectories of velocity and normal acceleration. A high correlation between the position and the loudness was observed for a flute sound.

Serafin et al. (2014) performed a user study on mapping for controlling physical modeling synthesis with data gloves. Participants, five experts and two novices, wore custom gloves with bend sensors and inertial measurement unit on both hands. They were presented eleven stimuli from different physical models and asked to perform gestures they would prefer to use for creating these sounds. Results showed several similarities in the gestures, especially controlling the pitch with the elevation of the associated hand.

Tanaka et al. (2019) use the Myo armband in a research project on continuous sonic interaction. Sounds can be connected with gestures in a user driven machine learning process. A custom Max/MSP abstraction connects the input device with the Wekinator and the sound synthesis processes in Max/MSP. The system allows *static regression*, which decomposes a gesture into breakpoints or full temporal trajectories, modeled by Hierarchical Hidden Markov Models (HHMM). After a set of sounds is designed with any synthesis algorithm, the corresponding gestures and postures are recorded and the system can be trained. The trained model can then be used in performances.

Visi, Caramiaux, and McLoughlin (2017) collect gestural data from a group of performers, in order to train a mapping model. At the beginning, six sound stimuli are generated using a waveguide-based physical flute model in Max/MSP, driven by 19 synthesis parameters. These sounds are then played back to eight participants, who were instructed to *mime a performance*

with a mute clarinet. Motion and orientation of instrument and performer, respectively the performer's limbs, are tracked with a camera-based *OptiTrack Flex 3*, alongside EMG data from Myo armbands. Three versions are recorded per participant and stimulus. All multi-modal motion performances are recorded and compared, using dynamic time warping for alignment.

Part II

The GLOOO System

Components of the GLOOO System

The project with the working title GLOOO is based on the design and development of four basic components, namely the *sample library*, the *modeling software*, the *synthesis software* and the haptic *control device*. Combined with an existing spatial rendering software and a sound reproduction system, such as a loudspeaker setup or headphones, these components constitute a musical instrument for expressive spatial sound synthesis.

Figure 5.1 shows the interaction of these elements in the complete analysis-synthesis system. The sample library is used in the modeling stage to generate a data set for the synthesis software. Audio output of the synthesis stage is then routed to a third party spatial rendering software. Both synthesis and spatialization are controlled with the BINBONG, the haptic musical interface developed for this application. All four components developed within this project will be introduced in dedicated parts of this Part II.

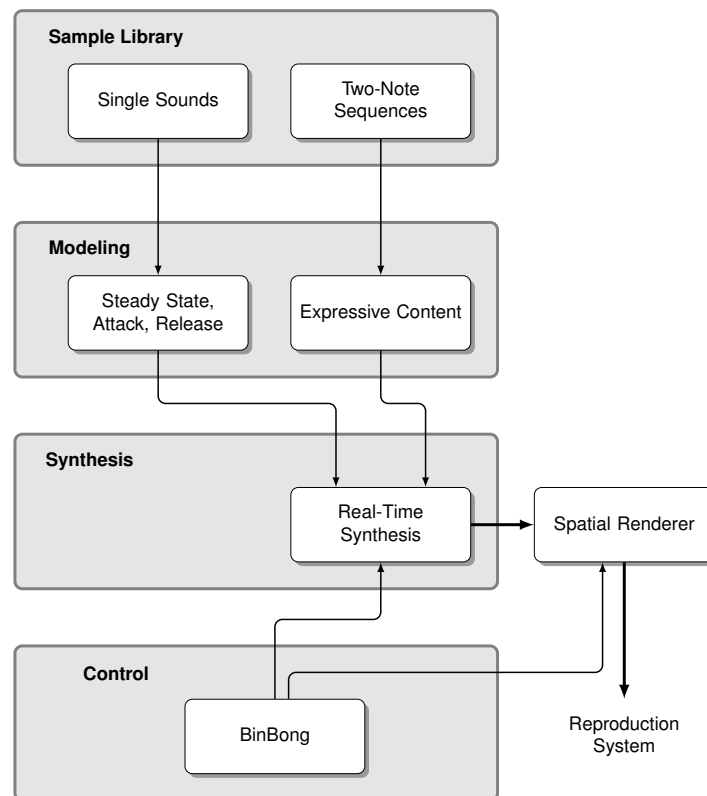


Figure 5.1: GLOOO project overview with the four components developed in this thesis and the third party rendering software.

Sample Library

This chapter covers the sample library which has been produced for the modeling algorithm. A brief introduction to the problem of finding a suitable library and the resulting motivation is given in Section 6.1. The structure of the TU-Note Violin Sample Library with its contents is explained in Section 6.2. Details on the recording setup and procedure are included in Section 6.3. Section 6.4 outlines the manual segmentation process which is necessary for using the library with modeling approach.

6.1 Libraries for Production and Research

A sample library of musical instrument sounds is an organized collection of sound recordings, suited for the use in sample based electronic musical instruments. Basic libraries for melody instruments usually capture the full pitch range of an instrument at different intensity levels, often with three to four notes per octave and two to three intensity levels (Lindemann, 2001). Section 3.2 explains the limitations of sampling synthesis in expressive musical applications and discusses solutions based on signal processing. Premium libraries, as for example the *EastWest Quantum Leap* (East West Communications, Inc., 2020) or the *Vienna Symphonic Library* (Vienna Symphonic Library GmbH, 2020) come with recordings of various articulation styles and extended playing techniques. *Native Instruments* lists a large variety of third-party sample libraries for their flagship sampler *Kontakt* (Native Instruments, 2020). By providing as many gestures as possible, such libraries extend the expressive capacities of sample-based instruments without involving complex manipulation algorithms and are thus state of the art in many use cases. However, tedious editing may be involved for reaching the desired result.

A valuable free sample library of different orchestra instruments is provided by Philharmonia Orchestra London (2020). It contains single sounds and short phrases of woodwind, brass, percussion and string instruments in up to four pitches per octave and four dynamic levels. Although the download offers MP3 format only, the samples are of decent quality.

Sample libraries are also an important resource for music research, especially in timbre research or in the development of algorithms for analysis-synthesis systems. Although commercial libraries can be used to some degree, copyright issues may prevent the publication of research based on them (Goto, Hashiguchi, Nishimura, & Oka, 2002). In addition, documentation of the recording procedure is usually not available which is disadvantageous and recorded sounds may be heavily processed. When it comes to best practices for the creation of sample libraries for music or research applications, literature is sparse. Companies selling high quality libraries usually do not share their trade secrets. Books dealing with the topic (Cann & Rausch, 2007; McGuire, 2007) are usually addressed to musicians and sound designers, not to the research community. Although a large number of sample libraries is available, research projects thus tend to generate their own, individual set of recordings. Often, the underlying research question demands very specific content and metadata.

The *RWC Music Database* (Goto et al., 2002; Goto et al., 2004) has been designed for the use in music information retrieval tasks. With a focus on performance analysis, the first version includes 215 solo and ensemble pieces from the genres popular, classical, and jazz. The second version features 315 recordings. The RWC comes with MIDI ground truth data, making it suitable for training and evaluating algorithms for many music information retrieval tasks, such as automatic segmentation and music transcription. The *McGill University Master Samples* (MUMS) is a research database of single instrument sounds, used in instrument recognition and classification tasks, sound synthesis and manipulation studies and perceptual experiments (Eerola & Ferrer, 2008). The MUMS includes 6546 sound samples, featuring instruments from the string, keyboard, woodwind, percussion and brass families, with 29 samples per instrument.

A different approach, trying to facilitate the use of copyright material in music information retrieval tasks, is proposed with the CMMSD, the *Commercial Monophonic Music Segmentation Database* (von Coler & Lerch, 2014). A set of solo passages from classical music recordings is compiled from Amazon downloads, listing only their unique download label. The accompanying hand labeled segmentation ground truth is published as a software repository on SourceForge. In this way, interested parties can purchase the musical content and download the metadata, separately. However, the library is designed for different research questions and can thus not be used in this project. But the segmentation guidelines and tools are the foundation for sample library presented in this section.

6.2 Audio Content

As explained in detail in Chapter 7, the GLOO project has specific requirements to the sample library used for the analysis stage. Existing libraries did not meet the requirements in terms of recording technique or metadata. Therefore, a custom data set, the *TU-Note Violin Sample Library*, has been recorded (von Coler, 2018). It is freely available under a Creative Commons license in a static repository with additional metadata (von Coler, Margraf, & Schuladen, 2018).

The *TU-Note Violin Sample Library* is made up of three parts, namely the *single sounds*, the *two-note sequences* and the *solos*. The 336 single sounds are used for learning the timbral characteristics of the instrument, by sampling combinations of different pitches and intensities. These statistical timbral qualities are the central data structure for the synthesis algorithm. Additionally, a structured set of 344 two-note sequences is designed to capture different articulation styles and note transitions as key factors of expressive musical content. A third part consists of scales in different specified versions of expressivity with a total of six items, as well as six free interpretations of solo pieces, intended to include additional expressive gestures.

In the scope of this thesis, the single sounds and the two-note sequences are used for different purposes in the modeling process, as visualized in Figure 7.1. They will thus be introduced in the following sections. The solo part of the library is intended for future uses.

6.2.1 Single Sounds

In the manner of standard musical sample libraries, the single sounds capture the instrument in different combinations of pitch and intensity, referred to as support points. Since this part of the library is used for creating a timbre model of the instrument, the number of support points has an influence on the accuracy of the model. The two-dimensional space is sampled in semitones across the full frequency range of the violin, including doubling through overlapping strings. **G**, **D** and **A** string are all sampled with 18 positions each, whereas the **E** string is sampled with 30 positions to the highest possible tone. All together this results in

$$3 \cdot 18 + 30 = 84 \text{ positions.} \quad (6.1)$$

Each position is recorded at four dynamic levels, spanning the maximum possible dynamic range of the violin. According to Meyer (2008), this interval covers a range from 58 . . . 99 dB SPL on the violin. The proposed levels are defined as **pp** - **mp** - **mf** - **ff** for the instructions. Combined with the number of positions, this leads to a total of

$$4 \cdot 84 = 336 \text{ samples.} \quad (6.2)$$

The resulting two-dimensional space is further referred to as *timbre plane*, well aware of the contrary term *timbre space* (D. L. Wessel, 1979) and the fact that timbre is usually defined by all features of sound excluding pitch and loudness. Nevertheless, this feature space of pitch and intensity is used to capture the timbre of the instrument as a function of these input variables. Figure 6.1 visualizes the timbre plane with the support points obtained through the sample library.

Pitch is further denoted with MIDI pitch values, also in the synthesis system. A list with all single sounds and their basic properties is include in Appendix I.

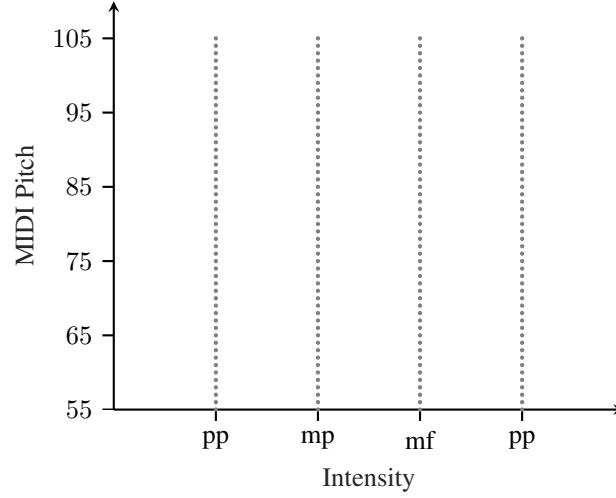


Figure 6.1: The timbre plane with a visualization of the support points.

In order to use the samples for generating a timbre model of the instrument, the single tones are all played without any expressive intention, with a focus on a long, steady sustain part. The resulting sounds do not contain any vibrato or accentuation and are considered the acoustical fingerprint of the instrument. Although tempo and note values are indicated to define a minimum duration, each note should be played as long as possible, using the complete bow. This is a delicate task, even for a professional violin player. Especially for high notes at low dynamic levels it is hard to maintain a steady sustain under these circumstances. Each item is recorded as a two shot sample, using one upbow and a downbow, clearly separated by a pause between. Takes were repeated when either the musician or the recording team was not satisfied with the result. The violinist was told to chose her natural bowing style for the recordings.

6.2.2 Two-Note Sequences

The two-note sequences are designed for modeling particular expressive musical content of violin play. According to the discourse in Section 2.1 this includes vibrato and selected articulation styles for the scope of this project. In order to capture these aspects for the entire instrument, the sequences systematically cover a feasible range of these expressive means in different pitch regions and dynamic levels. Each sequence is composed of two consecutive notes. Taking all possible note transitions between two notes on the violin is not feasible. With only 18 positions on four strings this results in

$$N = \binom{n}{k} = \binom{(4 * 18)}{2} = 2556 \quad (6.3)$$

combinations of different positions plus 72 repetitions of the same position. Since non-repetitive transitions can be played in two directions — upwards and downwards — these combinations are even doubled, resulting in 5184 sequences. This set is too large for recording or a systematic

exploration. For this reason, a sub set of five transitions is defined, spanning a wide range of the possibilities.

The five combinations captured in the two-note section of the library are visualized in Figure 6.2. They include different intervals, pitch regions and string combinations. The four intervals *fifth - one string*, *fourth - low*, *fourth - high* and *repeat* can be applied on all four strings, the remaining *fifth - two strings* only three times between the four strings. All combinations with exception of the *repeat* are played both upwards and downwards.

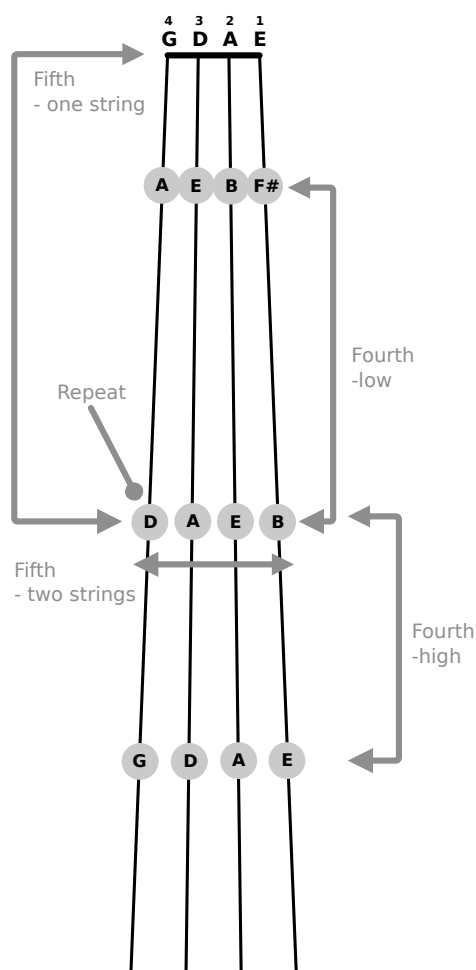


Figure 6.2: Violin fingerboard with intervals captured in the two-note section.

All two-note transitions are played in six different expressive versions, defined by the articulation style and the amount of vibrato. The three common articulation modes *Détaché*, *Lagato* and *Glissando* are included, each played with and without vibrato, as shown in the combination matrix in Table 6.1.

In addition, all possible variations are recorded in two dynamic levels, namely **mp** and **ff**. Some combinations between transition and expressive style are not applicable. The repeated notes, for

Table 6.1: Combination matrix for the different expressive styles to be recorded.

	Détaché	Legato	Glissando
No Vibrato			
Vibrato			

example, have no direction and the fifth on two strings allows no glissando. Altogether, the two-note section contains 344 items, all listed in Appendix II with their basic properties. Each note is held for a total of about of two seconds, resulting in approximately 5 seconds per sequence, considering the decay of the sounds.

6.2.3 Solo

The solo section of the sample library contains scales and excerpts from compositions. Both are intended for modeling further expressive content and have not been used in the course of this thesis. Nevertheless, their content will be briefly explained.

Scales

Two scales are included, namely a minor scale played downwards from ISO pitch **A5** and a major scale played upwards from ISO pitch **C4**. Both are recorded in three expressive styles, resulting in the combinations shown in Table 6.2. The first version, *plain*, is supposed to be performed without any specific expression, *expressive 1* with medium expressive gestures and *expressive 2* with emphasized expression. The interpretation of these three descriptions was left to the performer.

Table 6.2: Library items as combinations of the two scales with three expressive levels.

	Major (up)	Minor (down)
plain	Scale 01	Scale 04
expressive 1	Scale 02	Scale 05
expressive 2	Scale 03	Scale 06

Compositions

For future purposes, a set of solo pieces is included in the library. These recordings can be used as a corpus for learning expressive gestures, similar to the method proposed by Lindemann (2007). In order to use the content of solo performances for modeling expressive musical content as defined within this project, they should meet the following requirements:

- monophonic only
- only standard techniques

- few/no staccato notes

Based on these requirements, the violin player picked six solo works from her active repertoire, listed in Table 6.3. Hence, she was familiar with the material. Among the recordings are dedicated solo works as well as voices extracted from pieces with other instrumentation.

Table 6.3: Compositions included in the solo part of the sample library (von Coler, Margraf, & Schuladen, 2018).

Solo Item	Composition	Composer
07	Sonata in A major for Violin and Piano	César Franck
08	Violin Concerto in E minor, Op. 64, 2nd movement	Felix Mendelssohn
09	Méditation (Thaïs)	Jules Massenet
10	Chaconne in g minor	Tomaso Antonio Vitali
11	Violin Concerto in E minor, Op. 64, 3rd movement	Felix Mendelssohn
12	Violin Sonata No.5, Op.24, 1st movement	Ludwig van Beethoven

6.3 Recording Setup and Procedure

6.3.1 Environment

Since reverb has a significant negative impact on the estimation of the fundamental frequency (Arroabarren, Zivanovic, Bretos, et al., 2002) and on partial tracking for spectral modeling, a recording space with minimal reverberation is essential for this sample library. For this reason, the anechoic chamber at the *Staatliches Institut für Musikforschung* (SIM) Berlin has been chosen. The spring-mounted room-in-room design has a volume of $V = 50 \text{ m}^3$ and an absorption coefficient of $\alpha = 0.99$ above a cutoff frequency of $f_c = 100 \text{ Hz}$ (Staatliches Institut für Musikforschung, 2020).

Two microphones were used for capturing the instrument, namely one *DPA 4099* cardioid clip microphone and one *Brüel & Kjær 4006* omnidirectional small diaphragm microphone with free-field equalisation. The DPA microphone was mounted on the violin, as shown in Figure 6.3, above the lower end of the f-hole in 2 cm distance. Due to the fixed position, movements of the musician do not influence the recording. The B&K microphone was placed with a boom in 1 m distance above the instrument, at an angle of approximately 45° , as shown in Figure 6.4. All material was captured in a sampling rate of 96 kHz and a depth of 24 Bit, using a RME Fireface UFX.



Figure 6.3: Position of the DPA microphone on the violin.

Michiko Feuerlein, a professional violinist, was hired for the recordings. She played a violin with Joseph Guarneri label, at a tuning frequency of 445 Hz. All single sound items were recorded in a one day session, followed by a second day for two-note sequences and solo items.



Figure 6.4: Michiko Feuerlein performing in the anechoic chamber with the B&K microphone in the upper right.

6.3.2 Instructions

In order to allow a smooth procedure and avoid confusion or missing items, all single sounds, two-note sequences and scales were translated to minimal score snippets with *LilyPond* (Nienhuys & Nieuwenhuizen, 2003) inside \LaTeX . Table 6.4 shows the set of marks used to generate all instructions. Strings were encoded using the numbering scheme presented in Table 6.5. All instructions were beforehand reviewed by the violinist.

Table 6.4: Expressivity marks used in the score snippets.




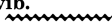

Play on string number	②
Play with Glissando	
Upbow	
Downbow	
Avoid any Vibrato	no vib.
Play with Vibrato	vib. 
Let the note decay completely and keep it completely separated from the following note.	

Table 6.5: String numbers used in the instructions.

String	Number
G	4
D	3
A	2
E	1

Figure 6.5 shows examples for a single sound item and a two-note sequence. The micro-scores of all 692 items in the full sample library are organized in a document with 62 pages. LilyPond and \LaTeX allowed an efficient programming and rendering of all items.

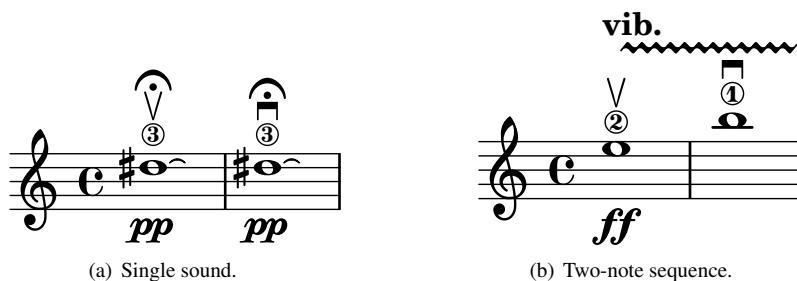


Figure 6.5: Micro-scores from the recording instructions.

6.4 Segmentation

6.4.1 The Note-Transition-Rest Paradigm

Before notes and transitions can be analyzed separately, a highly accurate segmentation process is needed. For the proposed modeling approach, recordings not only need to be segmented into separate notes, but into smaller units, such as steady-state and transient portions of sounds. Such procedure is also referred to as *intra-note-segmentation*. This is still a rather simple task compared to other problems of music information retrieval, especially with the high quality recordings from a non-reverberant environment. However, the required level of accuracy and robustness make it challenging. The statistical modeling algorithm, introduced in Chapter 7, can only be applied after a robust and accurate pre-processing and segmentation.

Since a similar kind of segmentation is a problem of many analysis-synthesis systems, several methods for automatic and supervised segmentation have been proposed within the related projects introduced in Section 3.4. Hahn (2015, p. 55) proposes a segmentation algorithm based on a threshold procedure in the energy trajectory. Extreme values are determined in the second derivative of the energy trajectory, in order to perform a *intra-note-segmentation*. A similar segmentation has also been proposed by Maestre et al. (2009). Within the reconstructive phrase modeling system, Lindemann (2001) also performs a segmentation of musical phrases into *attack*, *release*, *transition* and *sustain* segments. The patent does not further explain the segmentation procedure.

Although the segmentation of monophonic music belongs to the simple tasks in music information retrieval, error-free results can not be taken for granted. In the preparatory work for this thesis, a complete machine learning approach to this problem was developed (von Coler, 2013). The blind segmentation algorithm, based on a hidden Markov model and a feature selection stage, delivered promising results. However, optimizing it would have resulted in a shift of focus towards machine learning, away from the actual scope of sound synthesis and instrument design. Even a highly optimized solution would need a review process to eliminate remaining errors. Hence, the sample library used in this work has been manually segmented, with the help of colleagues and students. The short burst of effort concludes the segmentation issue and provides a conclusively labeled data set which can also serve as a ground truth for segmentation algorithms and other music information retrieval tasks.

For the requirements of the statistical modeling approach, the *note-transition-rest* paradigm is proposed as a combined method for inter- and intra-note level segmentation. Within this paradigm, single sounds and musical phrases are first split into segments from three categories. The *note* category contains only sustain segments of the sound, whereas the *rest* category covers all segments where the instrument is not sounding. The *transition* category encompasses all transitions between note and rest segments, including attack, release and glissando segments. In order to detect these categories, the proposed segmentation does not work on thresholds of specific feature trajectories but aims at a general distinguishability between stationary and transient parts, as implemented in the previously mentioned automatic approach (von Coler, 2013).

Regarding a separate analysis of attack, sustain and release portions of musical sounds, the *note-*

transition-rest paradigm has certain advantages. The differences between a segmentation method based on RMS threshold and a method relying on spectral features of the sound are illustrated for an attack segment of a violin sound from the library in Figure 6.6 and for the related release segment in Figure 6.7. End of attack, respectively beginning of release for the RMS based paradigm, are marked with a grey circle, whereas the gray area marks the duration of attack and release following a spectral criterion. Sounds of excitation continuous instruments show a significant increase after the actual attack transient has passed. Similarly, the energy usually decreases, before the actual release is initiated. Regarding their spectral features, these portions of the beginning and end of the note can rather be assigned to the sustain segment.

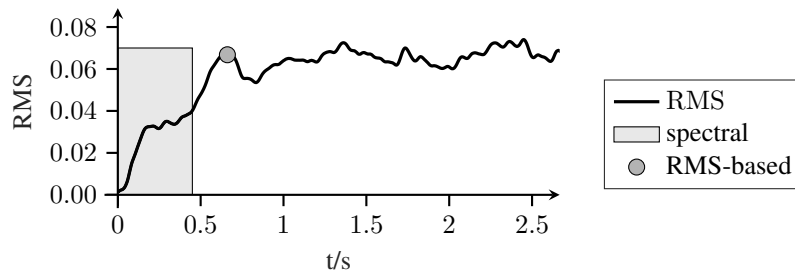


Figure 6.6: End of the attack segment through RMS threshold method and attack segment obtained through spectral criterion.

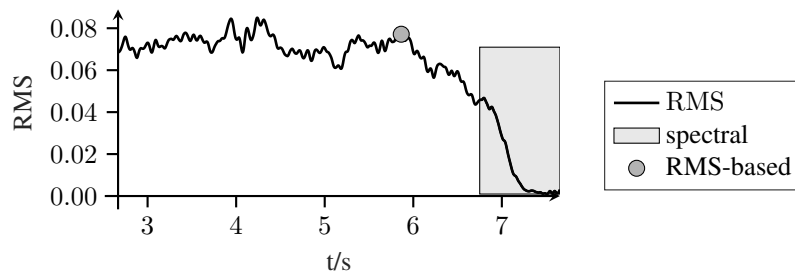


Figure 6.7: Beginning of the release segment through RMS threshold method and release segment obtained through spectral criterion.

6.4.2 Tools and Procedure

Based on preliminary work and the experience within this project, a trained individual can distinguish between steady state and transient parts based on representation of the waveform and the spectrogram, using the right tools. All items were annotated using a predefined layout in the software *SonicVisualiser* (Cannam, Landone, & Sandler, 2010), as shown in Figure 6.8 for a two-note sequence. The upper panel visualizes the energy contour, whereas the lower panel shows the peak frequency spectrogram, always using the same basic parameters of visual representation with slight adjustments for each item. All segments are already labeled in this example, indicated by the areas of different grey level.

A set of guidelines has been developed which allows a consistent labeling process with multiple individuals involved in the process (von Coler & Lerch, 2014; von Coler, 2018). First criterion

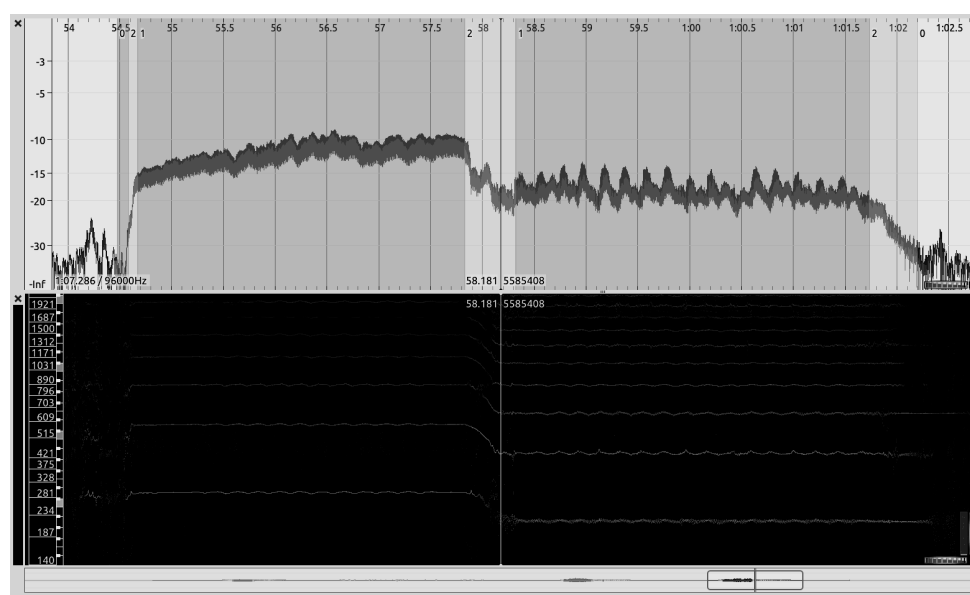


Figure 6.8: Segmentation setup in SonicVisualiser for two-note item 21 with vibrato and glissando.

for identifying transition segments is the harmonic to inharmonic ratio, which is visible in the spectral representation. Areas with prominent stable partials are labeled as notes, whereas transient segments in attack and releases, as well as in legato transitions show a significant amount of energy beside the partials' frequencies. The energy contour is used for confirmation and adjustment of the selections. Glissando transitions are localized in the trajectories of the partial frequencies, as shown in the lower panel of Figure 6.8. Finally, an auditive evaluation of the segmentation is possible in SonicVisualiser, through playback of the result at reduced playback rates with a sonification of the segment boundaries.

Figure 6.9 shows the logarithmic energy contour for the single sound item 333 of the single sounds with an ISO pitch of **A7** an intensity of **pp**. This combination of the highest pitch at lowest intensity is the most difficult case for both manual and automated segmentation. Clear segments are hard to determine due to the low signal to noise ratio and the slopes are less prominent. The related peak frequency spectrogram is visualized in Figure 6.10, with attack and release segments marked as dark gray areas. In this representation the transient portions of attack and release are still recognizable as areas of higher density in the spectrum.

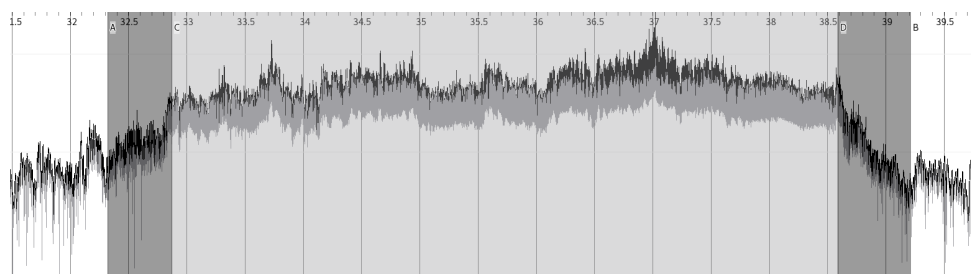


Figure 6.9: Segmentation of single note item 333 with SonicVisualiser the time domain.

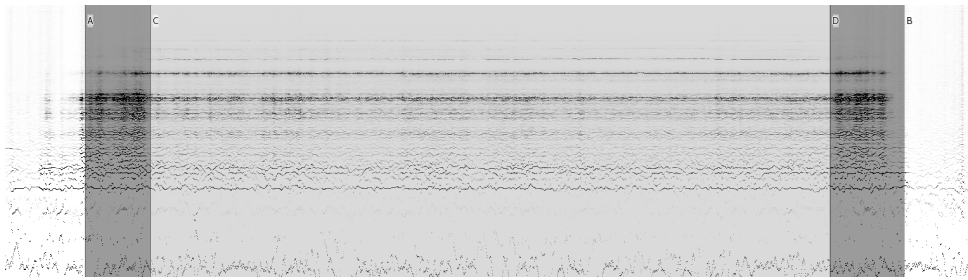


Figure 6.10: Segmentation of single item 333 with SonicVisualiser in the time-frequency domain.

6.4.3 Label Data

A dedicated segmentation file is created for each item in the sample library, with consistent labeling schemes for all files within the single sound and two-note sections. For every item in the single sound section, the segmentation file is formatted as shown in Listing 6.1, with the left column containing time instants in seconds and the right column the related labels. The attack segment is located between the markers A and C, the sustain segment between the markers C and D and the release segment between the markers D and B. Every two-note sequence is labeled with the structure shown in Listing 6.2. A 0 marks the beginning of a rest segment. The beginning of every transition is labeled with a 2. More details on their nature is included in the metadata in Appendix II. Beginnings of all note segments are labeled with a 1. Segmentation files for the solo items follow the same labeling scheme as the two-note sequences.

6.000666666	A
6.064000000	C
10.196250000	D
10.672500000	B

Listing 6.1: Single sound label file (item 56).

5.081250000	0
5.377500000	2
6.052500000	1
9.045000000	2
9.262500000	1
12.225000000	2
12.948750000	0

Listing 6.2: Two-note sequence label file (item 107).

Analysis and Modeling

Purpose of the analysis and modeling stage is the transformation of the audio content in the sample library into a data set for the use in the real-time software for statistical spectral synthesis. This chapter explains relevant aspects of this procedure in detail. A brief overview over the modeling process and the use of the individual sample library components is given in Section 7.1. Section 7.2 outlines the first step, the spectral modeling. This involves the partial tracking and the modeling of the residual component. The novel statistical modeling method of the thus obtained data is introduced in Section 7.3, followed by the transition modeling in Section 7.4. Finally, the data format and structure of the resulting model for the use in the synthesis system is presented in Section 7.5.

7.1 Overview

Figure 7.1 shows the analysis and modeling part of this project as a flow chart. The procedure, programmed in Matlab and accessible in a public repository (von Coler, 2020b), is fed with the single sounds and two-note sequences of the prepared sample library, as presented in Chapter 6. First stage of the analysis process is the spectral modeling, based on partial tracking and noise modeling. This procedure is performed in the same manner for the complete content of the sample library.

Segmentation is performed by splicing single sounds and two-note sequences, based on the annotations from the sample library. The following steps are individual for different segment types. Sustain portions of the single sounds are processed by the actual statistical modeling algorithm. Attack and release segments from the single sounds are separately modeled as exponential trajectories with statistical parameters. The glissando segments are taken from the two-note sequences and modeled as sigmoidal trajectories. Modeling results of all three segment types are then combined in the *synthesis model* and stored for later use.

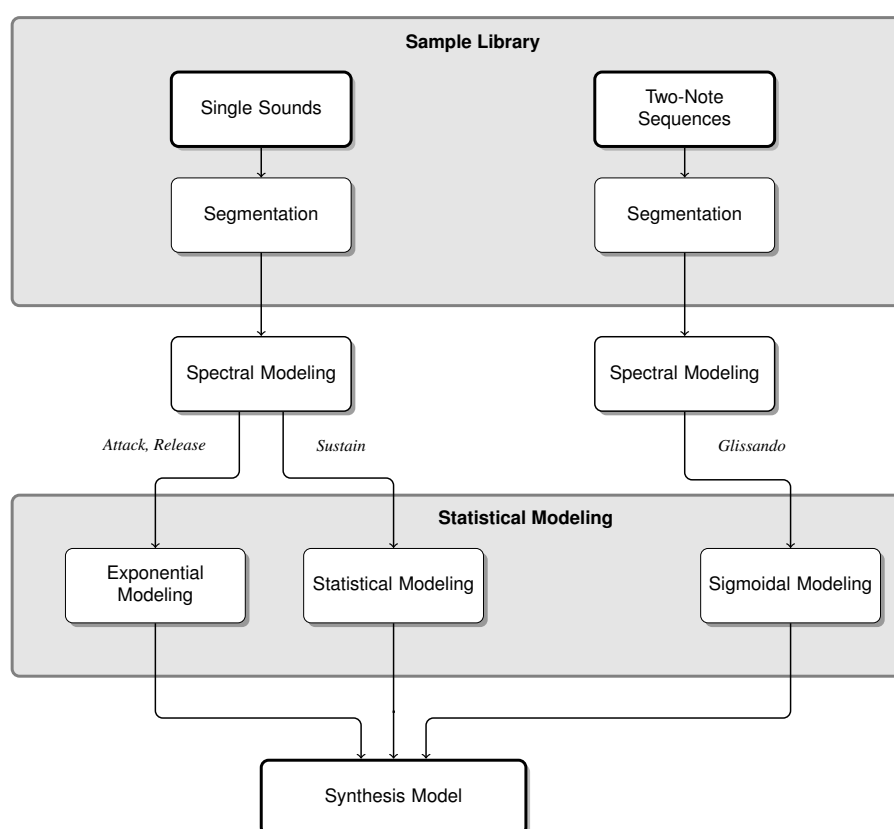


Figure 7.1: Block chart of the analysis and modeling process.

7.2 Spectral Modeling

7.2.1 Pitch Tracking

Pitch tracking is the first stage in the spectral modeling process. Various adequate algorithms for monophonic pitch tracking exist and the analysis environment allows the selection of different implementations. Among them are the established YIN algorithm (de Cheveigné & Kawahara, 2002) and the SWIPE algorithm (Camacho, 2007). More recent approaches, like the Kalman-Filter approach presented by Das, Smith, and Chafe (2017), are also considered but did not increase the robustness of the pitch tracking.

Although the case of monophonic analysis is simple compared to polyphonic analysis tasks, the results are still inaccurate when it comes to detailed analyses. All algorithms show drawbacks and benefits in certain situations, depending on the signal to noise ratio. Especially their mechanisms for dealing with regions of inconclusive pitch, caused for example by *pseudo polyphony*, affect the following partial tracking. Pseudo polyphony occurs when consecutive sounds of an instrument overlap and multiple pitches are present, although played in a monophonic way. In fast note transitions it might thus be beneficial to consider a partly polyphonic analysis, since a sound might “*carry over into the second note, blurring the transition*” (Strawn, 1986). This effect is enhanced by strong reverberant environments or instruments with large resonating bodies. Using the violin in the anechoic chamber these problems are almost negligible.

Different methods for the refinement of the f_0 -estimates exist (Erro et al., 2014). One is the use of the derivatives of the instantaneous phase, another one other is entitled *QHM-based frequency correction*. Erro et al. (2014) also propose the correction of the estimated fundamental frequency by averaging over all N corrected partial frequencies f_i , where w_i represents an adjustable weighting coefficient:

$$\Delta f_0 = \frac{\sum_{i=1}^N w_i \Delta \frac{f_i}{f_0}}{\sum_{i=1}^N w_i} \quad (7.1)$$

For the further analyses presented in this thesis, the SWIPE pitch trajectories are used at a sampling rate of 96 kHz, with a hop-size of 265 samples, respectively 2.7 ms. Limits for the fundamental frequency are $f_{min} = 20$ Hz and $f_{max} = 4000$ Hz. Using this basic algorithm, results can still be erroneous for certain parts of the samples in the library. However, since the statistical modeling approach inherently smoothes out such irregularities, suitable results are achieved without extensive refinements.

7.2.2 Partial Tracking

Since the sample library contains anechoic, monophonic material without rapid fluctuations and with a high signal-to-noise ratio, partial tracking can be performed based on standard algorithms. Starting point is the procedure presented by Serra and Smith (1990). It performs partial track-

ing based on the previously extracted fundamental frequency trajectories. A short-time Fourier transform (STFT) with fixed parameters is calculated. Hop size of the partial tracking is tied to the 2.7 ms hop size of pitch tracking. Based on this sampling rate, the STFT is performed with a window size of 2^{12} samples, zero-padded to 2^{13} samples. Windowing is performed with a Hann window of the according length.

Peak Estimation

For each time frame of the STFT, spectral peaks are detected at the estimated positions of the partials, if the signal shows a significant periodicity. In addition to the the fundamental frequency, the SWIPE algorithm calculates a measure for the pitch strength, ranging from 0 for noise without any periodicity to 1 for a purely harmonic signal. This information is used and peak detection is only performed above a threshold of 0.1 for the pitch strength.

Up to 80 peaks are measured, using a local maximum procedure within a defined search range for each partial. Thus, inharmonicity can be captured. The boundaries f_{min} and f_{max} of the search range for a peak of the partial at index i are set using the fundamental frequency f_0 :

$$f_{min}(i) = f_0(i - 1 + 2^{-\frac{6}{12}}) \quad (7.2)$$

$$f_{max}(i) = f_0(i - 1 + 2^{\frac{6}{12}}) \quad (7.3)$$

Local maxima determined through this procedure are refined using the quadratic interpolation technique, QIFFT (J. O. Smith & Serra, 1987). This results in decimal values instead of the integer values given by the support points of the Fourier transform. A safety check ensures that the interpolated peak does not exceed the boundaries of the STFT. The estimated partial frequency is calculated from the decimal position i^* , using the FFT size N_{FFT} and the sampling rate f_s :

$$f_i = f_s \frac{i^*}{N_{FFT}} \quad (7.4)$$

Phase Estimation

After amplitudes and frequencies have been estimated for every partial in an STFT frame, the related phase values are calculated using a subtraction method. Compared to methods based on the complex spectrum (McAulay & Quatieri, 1986; Serra, 1997), this approach is computationally intensive but robust and accurate. For each partial with the frequency f_i , the remainder x^* is calculated when subtracting it from the complete signal with the previously estimated frequency and amplitude values at different phases φ^* for an STFT frame with the length L :

$$x^*[\varphi^*] = \sum_{n=1}^L \left(x[n] - \sin(2\pi f_i \frac{n}{f_s} + \varphi^*) \right), \varphi^* = -\pi \dots \pi \quad (7.5)$$

The partial's phase is then obtained as the arg min of φ^* :

$$\varphi_i = \arg \min_{\varphi^*} \left[\sum_{n=1}^L \left(x[n] - \sin(2\pi f_i \frac{n}{f_s} + \varphi^*) \right) \right], \varphi^* = -\pi \dots \pi \quad (7.6)$$

Figure 7.2 shows a constructed example of x^* and the global minimum. In order to increase the accuracy, the subtraction method is performed iteratively. Once a local minimum is found, a second stage is applied between the local minimum and the lowest neighboring value. Using eleven points per stage, this nested approach results in an accuracy of

$$a = \frac{\frac{2\pi}{11}}{11} = 0.0519 \text{ rad.}$$

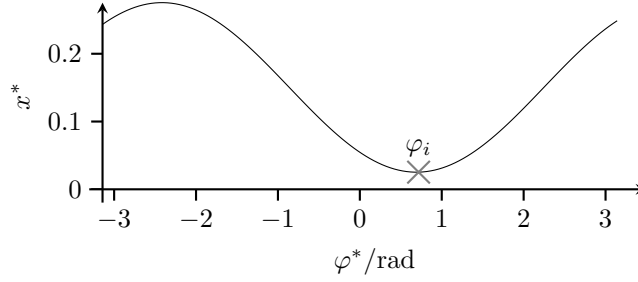


Figure 7.2: Estimating the phase φ_i of a partial by subtraction.

Partial Trajectories

In polyphonic or inharmonic signals, the connection of spectral peaks between consecutive frames can be demanding, since the trajectory continuation is ambiguous has to be estimated. This problem can be solved using prior trajectory values, for example with linear prediction methods (Lagrange, Marchand, & Rault, 2004; Lagrange, Marchand, & Raultt, 2005). For off-line applications, non-progressive algorithms find optimal partial trajectories on the full matrix off sinusoidal peaks (Bartkowiak & Zernicki, 2011). In case of the monophonic signals of this analysis, the trajectory continuation is a rather straightforward step, since the partial trajectories are linked to the fundamental frequency. Trajectories for amplitude, frequency and phase of each partial are explicit and can be obtained by concatenating the values at the relevant partial index.

Figure 7.3 shows partial amplitudes for the first 20 partials of a violin sound. Trajectories of lower partials have a higher mean amplitude are more steady then those of higher partials during the sustain segment. During the release segment, higher partials show a more rapid decrease in amplitude. Figure 7.4 shows the associated trajectories of partial frequencies. Apparent jitter in the partial frequency occurs when the partials have a very low amplitude, as observed for higher partials during the sustain. Noisy partial frequency trajectories occur in the release segment, after the partials' amplitude has dropped substantially and does not contribute to the sound. Figure 7.5 shows the unwrapped partial phases of the same sound. All trajectories are stored as tab-separated text files for each single sound and can thus be used in the following modeling steps.

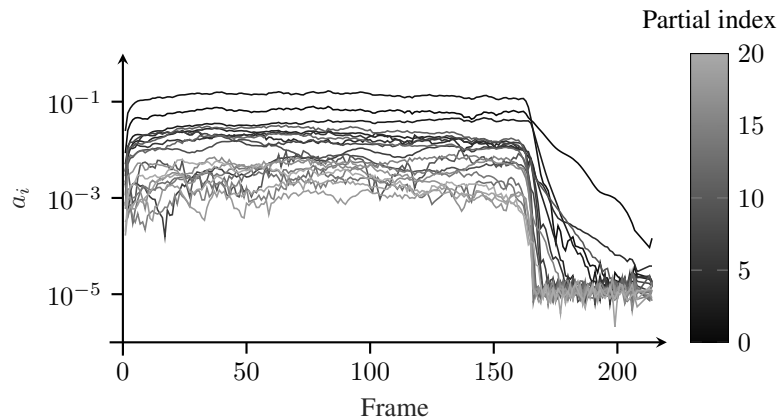


Figure 7.3: Amplitude trajectories of the first 20 partials for item 22 from the single sounds of the sample library.

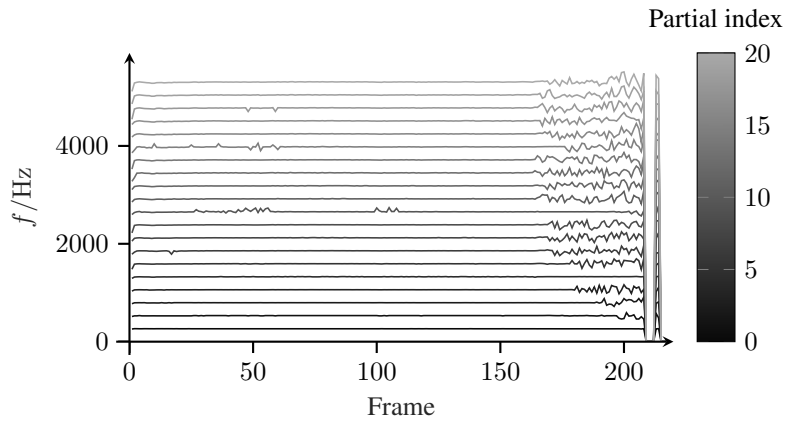


Figure 7.4: Frequency trajectories of the first 20 partials for item 22 from the single sounds of the sample library.

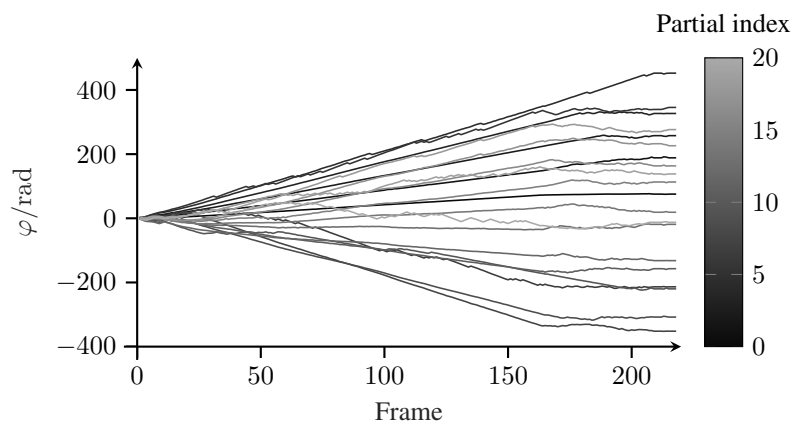


Figure 7.5: Unwrapped partial phases of the first 20 partials for item 22 from the single sounds of the sample library.

7.2.3 Residual Modeling

Based on the results of the previously described partial tracking, the harmonic signal can be resynthesized with the original amplitudes, frequencies and phases of the partials. The residual signal is then calculated by subtracting the resulting resynthesis from the original sound, according to the relations introduced in Section 3.3. Subsequently, the residual is modeled with a Bark frequency scale filter bank, with the cutoff frequencies listed in Table 3.2 of the introductory part. 6th order Butterworth bandpass filters are used with the frequency responses of the individual filters in the bank plotted in Figure 7.6.

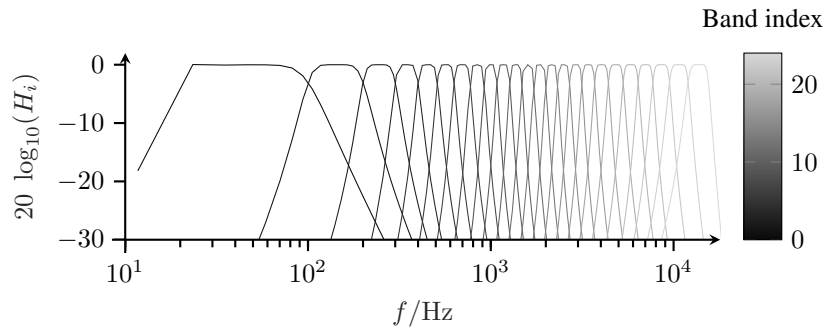


Figure 7.6: Amplitude responses of the individual IIR filters in the Bark frequency scale filter bank.

The residual components of all items in the library are processed with the filter bank, resulting in 24 band-filtered signals per sound. Subsequently, the energy envelope is calculated for all band-pass signals using a smoothed root mean square. Figures 7.7 through 7.11 show examples for resulting energy trajectories. In lower frequency bands, environmental noise and other noise sources in the recording chain outweigh the noise caused by the instrument. For frequency band four and upwards, the energy trajectories show significant stochastic fluctuations around a steady mean value during the sustain.

The residual signal obtained through subtraction is still correlated with the deterministic component, in consequence of modeling errors and the actual relation between the two parts. These correlations are smoothed out in the frequency domain when applying the band pass filters. The trajectories of all 24 noise bands are stored as tab separated text files for all items in the sample library for later use.

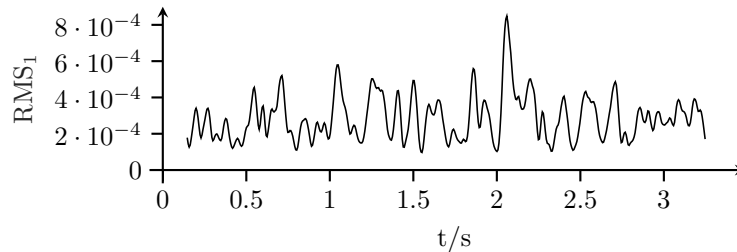


Figure 7.7: Energy trajectory of the first Bark band for item 08 from the single sounds of the sample library.

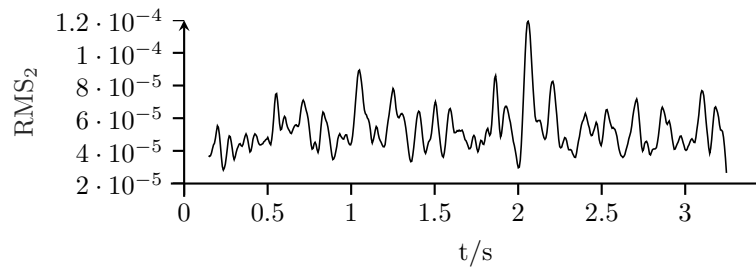


Figure 7.8: Energy trajectory of the second Bark band for item 08 from the single sounds of the sample library.

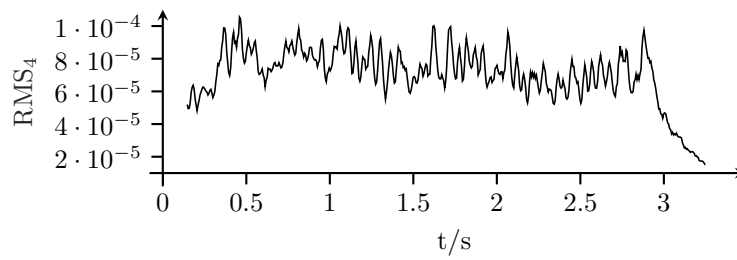


Figure 7.9: Energy trajectory of the fourth Bark band for item 08 from the single sounds of the sample library.

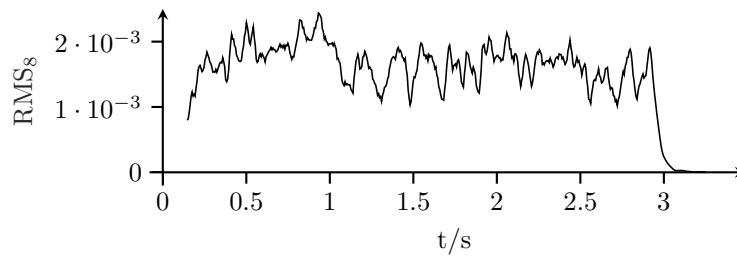


Figure 7.10: Energy trajectory of the eighth Bark band for item 08 from the single sounds of the sample library.

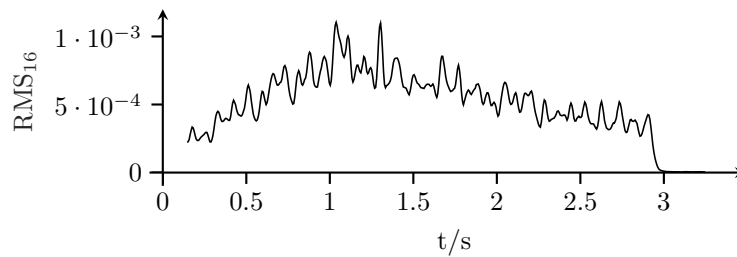
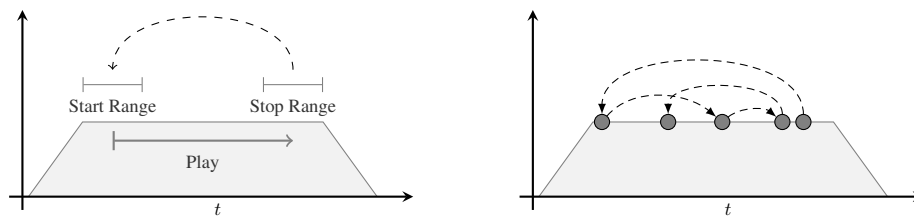


Figure 7.11: Energy trajectory of the 16th Bark band for item 08 from the single sounds of the sample library.

7.3 Statistical Modeling

The spectral modeling procedure outlined in the preceding Section 7.2 results in a set of parameter trajectories, describing the tonal and residual part for each item in the sample library. Synthesis based on this data set results in high quality sounds, to a certain degree indistinguishable from the original recording. At this point, however, the expressive means of the resynthesis are still limited, just as in sample playback. The exact same sound can be reproduced and this stage of spectral modeling can be considered a method of data reduction. However, the nature of the model makes it possible to manipulate the properties of the sound, including for example time stretching, pitch shifting and timbre manipulations. This requires further steps, which will be introduced in this section.

The statistical modeling procedure described in this section aims at providing means for extended manipulations of the spectral modeling data, based on expressive control gestures. It can be considered a method for “*removing the time axis*” (D. Wessel et al., 1998) from the spectral data by capturing the quasi stochastic fluctuations of parameters, which are not directly controlled by the control parameters. *Shimmer* and *jitter* describe such stochastic micro-fluctuations of the overall amplitude, respectively the fundamental frequency trajectory (Verfaillie et al., 2005). This concept can be extended to the parameters of single partials and thus result in rapid fluctuations of timbre. These stochastic modulations are an integral part of the sounds of musical instruments and contribute to their *richness and realism* (Lindemann, 2010). For many digital processes in musical applications, the integration of such stochastic components is an important step for achieving a certain naturalness or vividness. Within the source-filter model proposed by Hahn et al. (2010), for example, additive noise is applied to the deterministic spectral envelope in order to achieve rapid fluctuations. The approach presented in the following section aims at modeling the shimmer and jitter from the previously calculated partial trajectories.



(a) Looping a sound within flexible start and stop ranges.

(b) Randomly selecting portions of a sound for frame wise synthesis.

Starting point for the statistical modeling was the idea to use loop points for extending the recorded sounds, as applied in standard sampling technology. This technique is in particular applicable for sounds with a pronounced sustain segment, such as the violin. In order to reduce the repetitiveness of a sound sustained through looping, the loop points can be placed randomly within defined limits in each cycle. Figure 7.12(a) illustrates this concept for an arbitrary sound. Ranges for selecting random start points and stop points are located within the steady state of the sound. Moving along with this thought, a random point could be picked from the trajectories for each synthesis frame. As visualized in Figure 7.12(b), all random points would be located within the sustain part of the sound. Such a procedure can actually be realized with different synthesis paradigms, such as granular techniques or spectral modeling, like in the project presented in this

thesis. Resulting synthetic sounds would be composed of a random sequence of timbres included in the original sounds and each sequence would have individual characteristics. This idea concludes in the conversion of the temporal trajectories for all spectral modeling parameters into probabilistic functions, described as statistical spectral modeling, as explained in the following section. This allows to also synthesize timbral frames not included in the original sound and morphing between different timbres.

7.3.1 Trajectory Decomposition

Statistical spectral modeling is designed to capture the small fluctuations in the sound of musical instruments, which are beyond the direct control of the musician, yet important for the timbral qualities. This is realized by exploiting the statistical properties of the sinusoidal modeling parameter trajectories. The trajectories need to be pre-processed, in order to isolate these micro fluctuations from components which are directly influenced by the musician. The latter are controlled directly in the synthesis system. This results in an independent modeling of timbre and expressive musical content.

A similar decomposition approach is common in related projects, especially for fundamental frequency trajectories. Rossignol (2000) introduces the concept of a step-wise fundamental frequency trajectory $f_{0_{DC}}(t)$. It represents the average frequency value during a note. The fundamental frequency trajectory within a note is smoothed by subtracting the variations with a frequency higher than the maximum vibrato rate, referred to as $f_{0_{AC}}(t)$. In the analysis of the singing voice, Saitou et al. (2002) also distinguish between the fluctuations induced by vibrato and those with slower variations. Arroabarren, Zivanovic, and Carlosena (2002) use the term *intonation* for the fundamental frequency trajectory without vibrato. A decomposition into a slowly varying and a rapidly varying components has also been proposed by Lindemann (2007, 2010) in the reconstructive phrase modeling system.

Similar to the approach in this thesis, Ardaillon et al. (2015) suggest the decomposition of f_0 trajectories into four parts. The melodic component is comprised of $f_{0_{DC}}(t)$ with transition segments and their overshoots, similar to the intonation component. The vibrato component features the periodic modulations. A *phonetic component* is related to the pronunciation of voiced consonants. The *jitter* component includes random uncontrolled variations. A listening test showed that jitter and the phonetic component increased the perceived quality of synthesized speech.

In the preliminary work of this thesis, the *Slope Overload Memory* (SOM) algorithm for the decomposition of pitch trajectories into three components was proposed (von Coler & Röbel, 2011). In addition to the model of Rossignol, it distinguishes between the periodic modulations $f_{0_{mod}}$, such as vibrato and the corrections $f_{0_{cor}}$, slowly varying fluctuations below a frequency of 5 Hz:

$$\begin{aligned} f_0(t) &= f_{0_{DC}}(t) + f_{0_{AC}}(t) \\ &= f_{0_{DC}}(t) + f_{0_{cor}}(t) + f_{0_{mod}}(t) \end{aligned}$$

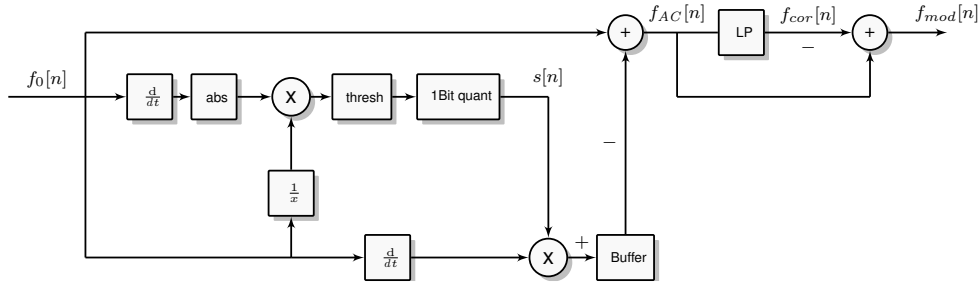


Figure 7.12: Flow chart of the SOM - f_0 decomposition algorithm (von Coler & Röbel, 2011).

A non-linear filter with a subsequent low pass has been proposed to perform this decomposition, as shown in Figure 7.12. The same method is used as a foundation for the trajectory decomposition stage in the GLOOO modeling process.

For the statistical modeling, the above introduced decomposition approach is extended to trajectories of the spectral modeling parameters. In order to isolate the rapid fluctuations, an additional stochastic component x_{rand} is thus added to the model. A parameter trajectory x is now composed of four components:

$$x = x_{AC} + x_{cor} + x_{mod} + x_{rand} \quad (7.7)$$

The piecewise constant x_{AC} is a steady component, including for example the average pitch or energy during a note. Corrections x_{cor} represent slow fluctuations around this value, usually below 5 Hz. Periodic modulations x_{mod} like vibrato are the most rapid parts of direct control with a maximum frequency of about 10 Hz. The stochastic component can hence be extracted from the trajectories through high pass filtering above this frequency. h represents the impulse response of a third order Butterworth high pass filter with a cutoff frequency of 10 Hz:

$$x_{rand} = x * h_{rand} \quad (7.8)$$

In fact, the statistical modeling approach processes the combination of the steady component x_{AC} with the rapid fluctuations x_{rand} of the spectral modeling trajectories. The remaining components can be directly influenced through control parameters.

The partial frequency trajectories of the single sound items need no pre-processing or decomposition prior to the statistical modeling, since the sounds are played as plain as possible. Figure 7.13 shows the normalized partial frequency trajectory during the sustain segment of item 40 in the single sound category. Figure 7.14 shows the probability density functions of the frequency trajectories for four partials during the sustain of the same sound. All partials show narrow distributions, indicating minimal deviations from the strictly harmonic model. Inharmonicity can thus be considered negligible for the violin under these conditions. The prominent extreme values for partial 3 and 4 indicate that the detected f_0 is lower than the actual fundamental frequency.

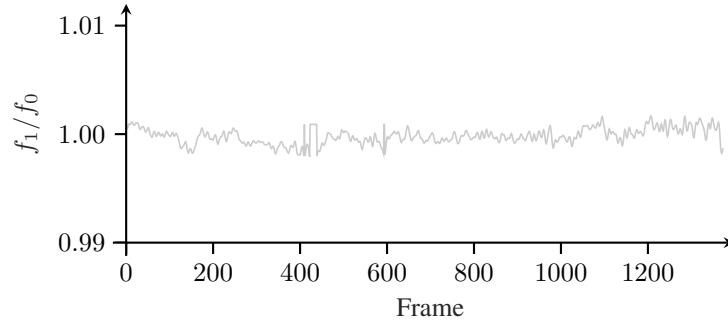


Figure 7.13: Normalized frequency trajectory for the first partial in the sustain part of single sound item 40.

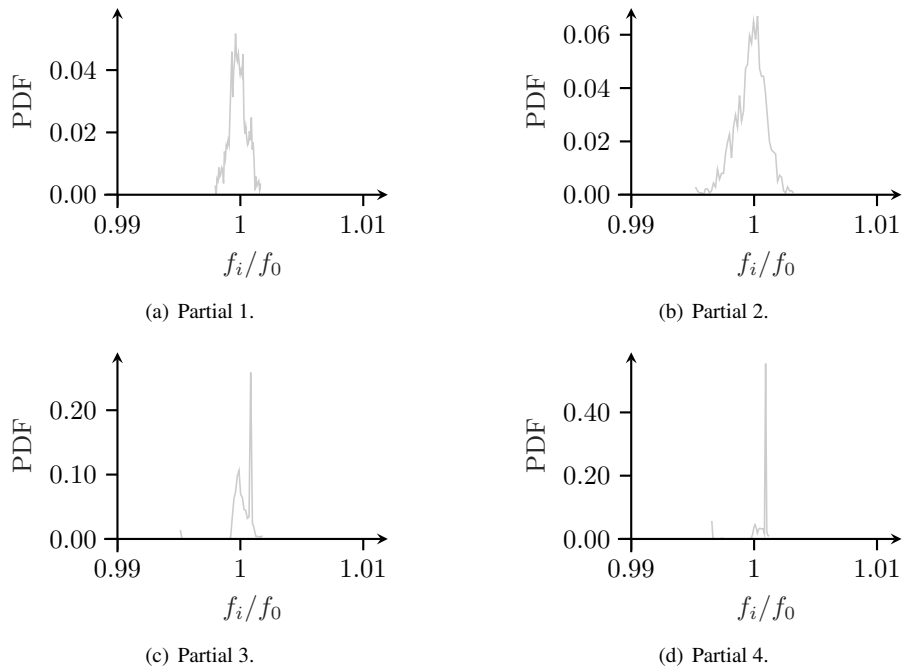


Figure 7.14: Probability density functions of partial frequencies during the sustain of single sound item 40.

Partial amplitudes show more prominent fluctuations during the sustain than the partial frequencies. Figure 7.15 shows the unprocessed trajectory a of the first partial's amplitude in single sound item 40 alongside the isolated components after filtering, as used in the statistical modeling. Figure 7.16 shows the related probability density functions of amplitude trajectories for the first four partials during the sustain, alongside the extracted relevant components $a_{DC} + a_{rand}$. All these plots show a significant narrowing of the distributions through the filtering, since the slowly varying modulations and corrections have been removed. The cutoff frequency in the decomposition is the crucial parameter for this compression effect. Resulting distributions should capture the characteristics of the fluctuations, discarding possible outliers which impair the quantization performed in the following step.

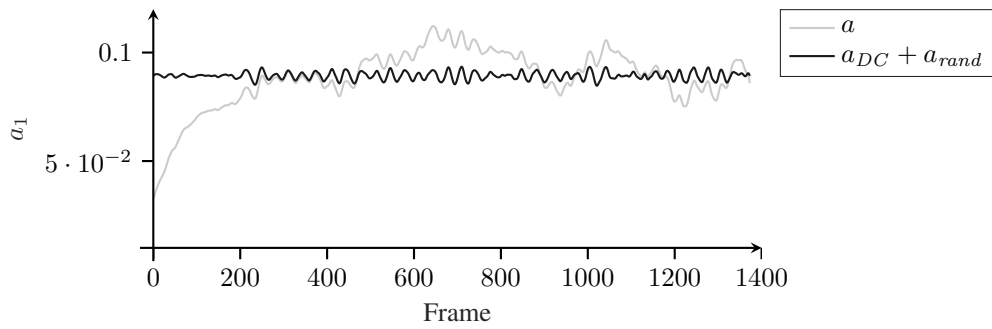


Figure 7.15: Decomposition of the first partial's amplitude trajectory for the sustain part of single sound item 40.

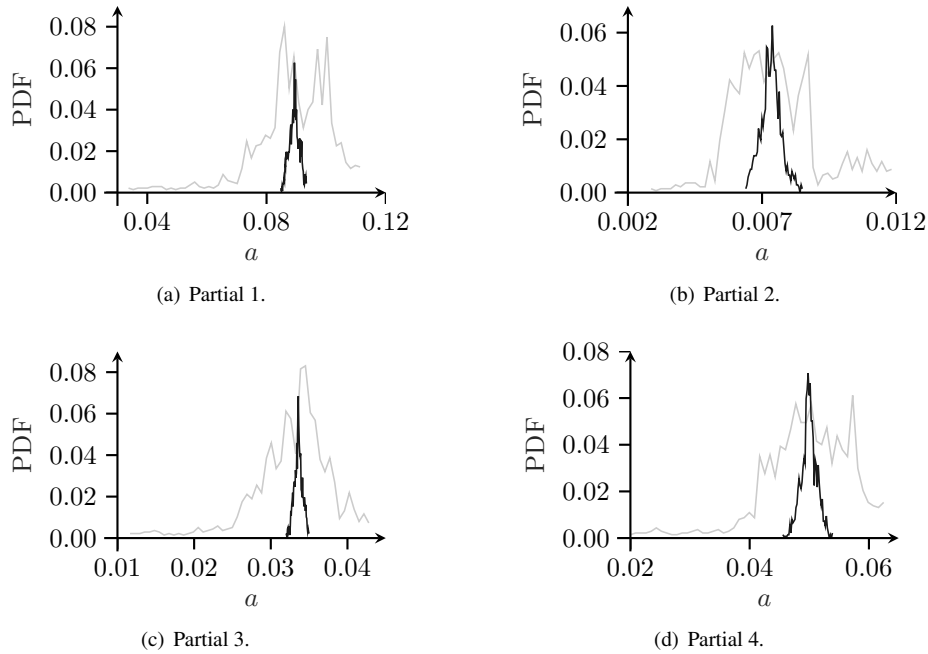


Figure 7.16: Probability density functions of partial amplitudes with separate plots for the complete trajectories (gray) and after decomposition (black).

Figure 7.17 shows the original energy trajectory of a Bark band together with the pre-processed version used for statistical modeling. Probability density functions are shown for energy trajectories before and after decomposition in Figure 7.18. Compared to the partial amplitude trajectories, the narrowing of the distributions through the filtering is less significant.

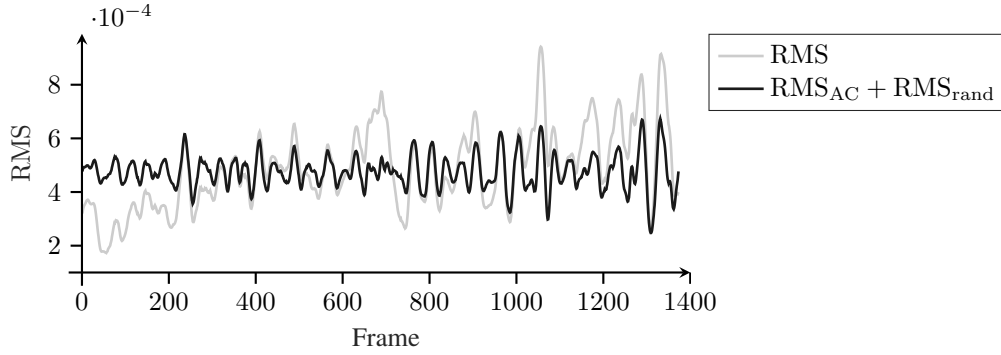


Figure 7.17: Decomposition of the residual energy trajectory for Bark band six in the sustain part of single sound item 40.

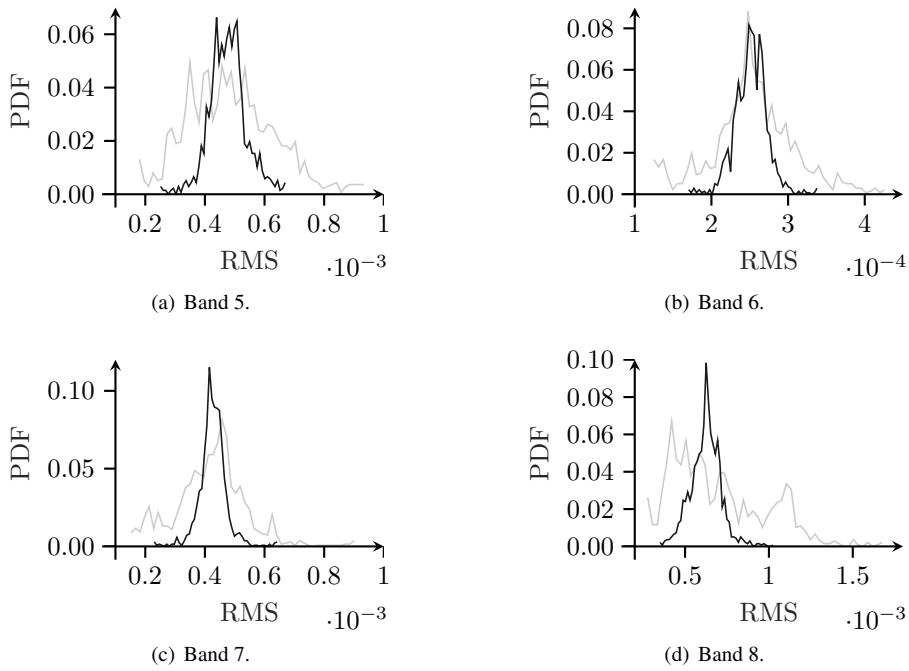


Figure 7.18: Probability density functions of Bark band energies for the original trajectory (gray) and after pre-processing (black).

7.3.2 Stateless Statistical Modeling

Probability Mass Function

Stateless Statistical Modeling is the basic way of removing the time axis through probabilistic means, proposed within this thesis. Trajectories $x[n]$ of all partial amplitudes and frequencies, as well as trajectories of the Bark band energies are therefore transformed into individual probability mass functions (PMFs). Only the sustain segments of the single sounds are processed in this step. Figure 7.19 shows a constructed example for a trajectory with normal distribution within the limits -1 and 1 .

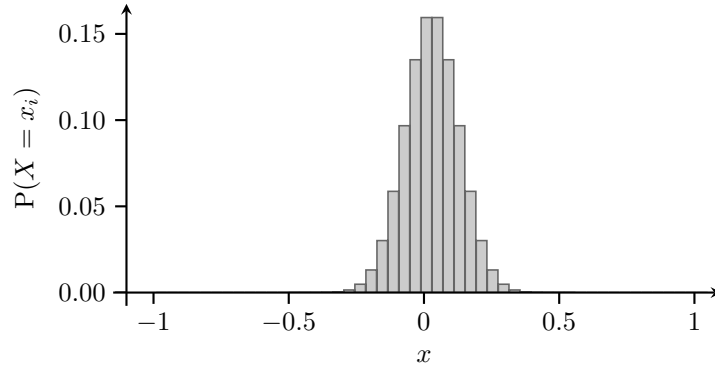


Figure 7.19: Example of a probability mass function (PMF).

The parameter values of the trajectories need to be quantized in order to obtain the discrete distribution functions. The boundaries for this step are defined by the basic statistical properties of the trajectory to be modeled. Assuming confined distributions for the feature trajectories after decomposition, upper and lower boundary are defined by the first percentile, in order to ignore outliers:

$$X_{min} = P(X < 0.01) \quad (7.9)$$

$$X_{max} = P(X > 0.99) \quad (7.10)$$

PMFs are calculated using a fixed set of $N_{support} = 21$ equidistant support points or quantization stages x_i :

$$x_i = X_{min} + i \frac{X_{max} - X_{min}}{N_{support}} \quad (7.11)$$

This number of stages results from preliminary tests and is directly influenced by the number of samples in the trajectory. More quantization stages result in a higher resolution, but also increase the likeliness of a sparsely populated distribution.

Values of the PMF are then calculated as the cardinality $|\cdot|$ of samples assigned to the related quantization stages x_i , after rounding all samples in $x[n]$ to the quantization grid. Absolute frequencies are normalized with the number of samples L :

$$\text{PMF}(i) = \frac{1}{L} \left| \{x[n] \mid x_i = \arg \min_X (|x[n] - X|)\} \right|, \quad n = 1 \dots L \quad (7.12)$$

Cumulative Mass Function

For an application in the synthesis algorithm, further processing of the above calculated distributions is necessary. In the next step, cumulative mass functions (CMFs), as exemplified in Figure 7.20, are calculated from the PMFs. This conversion can be realized with a straightforward accumulation formula:

$$\text{CMF}(i) = \sum_{x=1}^{x_i} \text{PMF}(x) \quad (7.13)$$

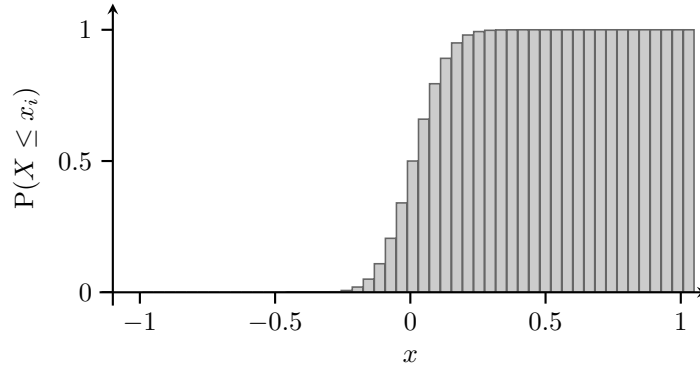


Figure 7.20: Example of a cumulative mass function (CMF).

Inverted Cumulative Mass Function

Finally, the inverted cumulative mass function $\text{ICMF} = \text{CMF}^{-1}$, also referred to as *quantile function*, is calculated. An example for a inverted cumulative mass function is shown in Figure 7.21. For the inverse transform sampling in the synthesis part, support points on the abscissa of the ICMF need to be equally spaced. The ordinate of the CMF is not equally spaced at this point. A set of N_p support points p_i is thus calculated in advance, equidistantly spanning the range between 0 and 1:

$$p_i = \frac{i}{N_p} \quad (7.14)$$

Single values of the ICMF are calculated at the support points p_i as the arg min of elements in the CMF smaller than p_i :

$$\text{ICMF}(i) = \arg \min_{x_i} (\text{CMF}(x_i) < p_i) \quad (7.15)$$

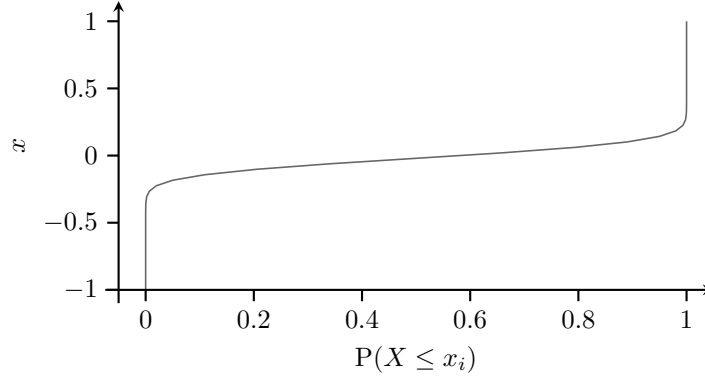


Figure 7.21: Example of an inverse cumulative mass function (ICMF).

Modeling Results

The statistical modeling procedure outlined above is applied to the trajectories of the partial amplitudes, partial frequencies and noise band energies for the steady state segments of all single sounds. Results are stored in tab-separated text files with 80 amplitude, 80 frequency and 24 energy trajectories per single sound item, alongside basic statistical properties of the distributions, like mean, median and standard deviation. This data set can later be used for different modes of statistical synthesis of the sustain segments.

For a better representation, example results of single sound item 55 are visualized as cumulative mass functions. Figure 7.22 shows the CMFs for the first 30 partial amplitudes. A decrease in mean energy is noticeable towards higher partials. Distributions for the first 30 normalized partial frequency trajectories are shown in Figure 7.23. Deviations from the strictly harmonic model are minute but systematic, since the mean frequency of higher partials decreases to about 1%. The higher the partial index, the lower the gradient of the CMF. Low partials thus have a steady frequency, whereas higher partials show a jitter in frequency of about 2%. This effect can be explained through inaccuracies in the measurement but also through friction between the bow and the string. CMFs for all 24 Bark band energies are shown in Figure 7.24. Gradients are noticeably low, representing the significant fluctuations in the energy trajectories.

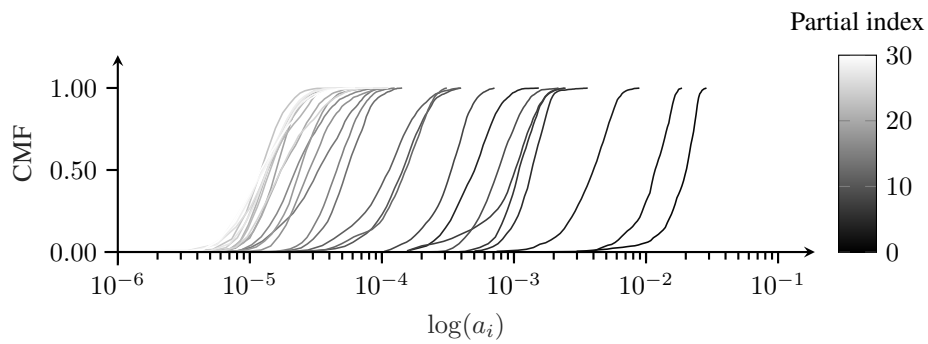


Figure 7.22: CMFs for the first 30 partial amplitudes of single sound item 55.

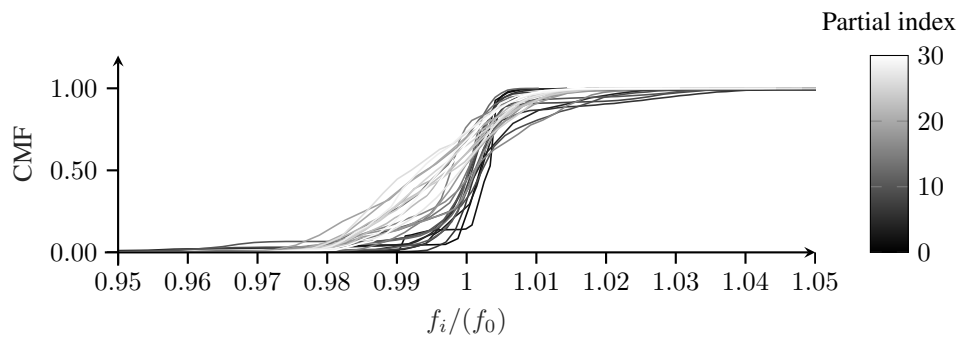


Figure 7.23: CMFs for the first 30 partial frequencies of single sound item 55.

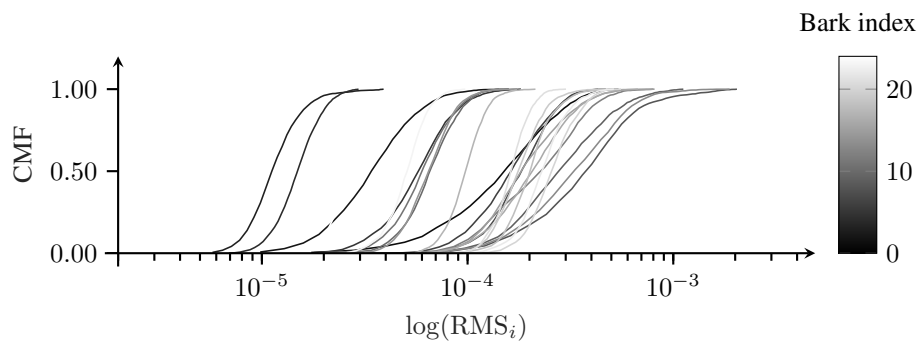


Figure 7.24: CMFs for the 24 bark energy trajectories of single sound item 55.

7.3.3 Markovian Statistical Modeling

The stateless approach for statistical modeling, introduced in the previous Section 7.3.2, captures the distribution properties parameter trajectories. However, their frequency characteristics are not preserved within that model. When used in a synthesis algorithm, fully random samples can be generated during each synthesis frame, similar to the principle illustrated in Figure 7.12(b). In doing so, discontinuities can occur in the synthesized parameter trajectories. As introduced later in Section 8.1.3, these discontinuities can be smoothed out, at the expense of altering the distribution. The *Markovian Statistical Modeling* algorithm presented in this section captures both the amplitude and frequency characteristics of stochastic processes. After quantizing the parameter trajectories to discrete states, they are therefore modeled as first order Markov processes (Ross, 2007; Alpaydin, 2010). During synthesis, past values can thus be taken into account when generating new random values.

Quantization

Since this statistical modeling approach is based on a discrete Markov process, quantization of the probability density functions is carried out as in the stateless modeling approach. $N_{step} = 21$ stages have been chosen for the modeling within this thesis, equally distributed between the minimum and maximum values of the pre-processed trajectories. Figure 7.25 shows the resulting probability mass function for the second partial's amplitude in the sustain segment of single sound item 01. The function is a slightly skewed normal distribution.

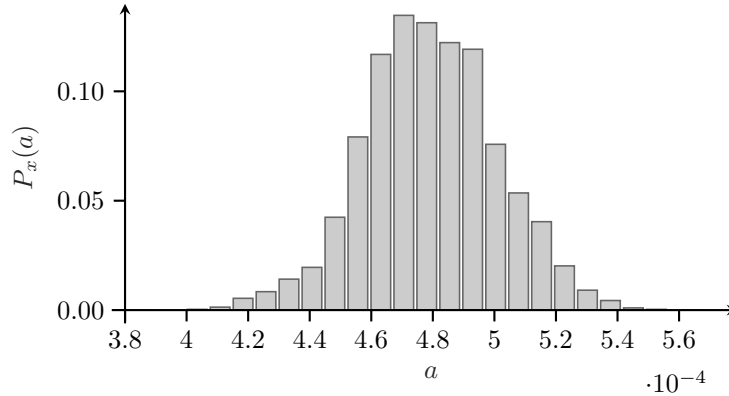


Figure 7.25: Probability mass function of the pre-processed amplitude trajectory for the second partial of single sound item 01 with 21 quantization stages.

Transition Probabilities

Each of the N_{step} quantization stages represents a state within a Markov process, used for modeling the temporal properties of the trajectory. In Markov processes, *transition probabilities* are calculated which encode the likelihood of transitions between the states, respectively the quantization stages. Individual transition probability functions PMF_i are calculated for each quantization step x_i by counting and normalizing the transitions to all states j for the trajectory

of the length L , resulting in the matrix $\text{PMF}(i, j)$ of transition probabilities:

$$\text{PMF}(i, j) = \frac{1}{L} |\{x[n] \mid x[n+1] = x_j\}|, \quad n = 1 \dots L \quad (7.16)$$

The resulting transition probabilities for one synthesis parameter of one sound can be combined to a transition matrix. This matrix fully describes the statistical properties of the quantized trajectory. Figure 7.26 shows the individual transition probabilities as probability mass functions for the first partial's amplitude of single sound item 01 in a surface plot. Transitions between states i are only possible if their distributions overlap. Distributions of neighboring quantization stages overlap along the diagonal, making transitions between nearby stages more likely and excluding transitions between distant quantization stages, thus limiting the bandwidth of the fluctuation. Distributions close to the average value are wider than those located at the boundaries. The out-most quantization stages show narrow, quasi uniform distributions, pulling the signal towards the average.

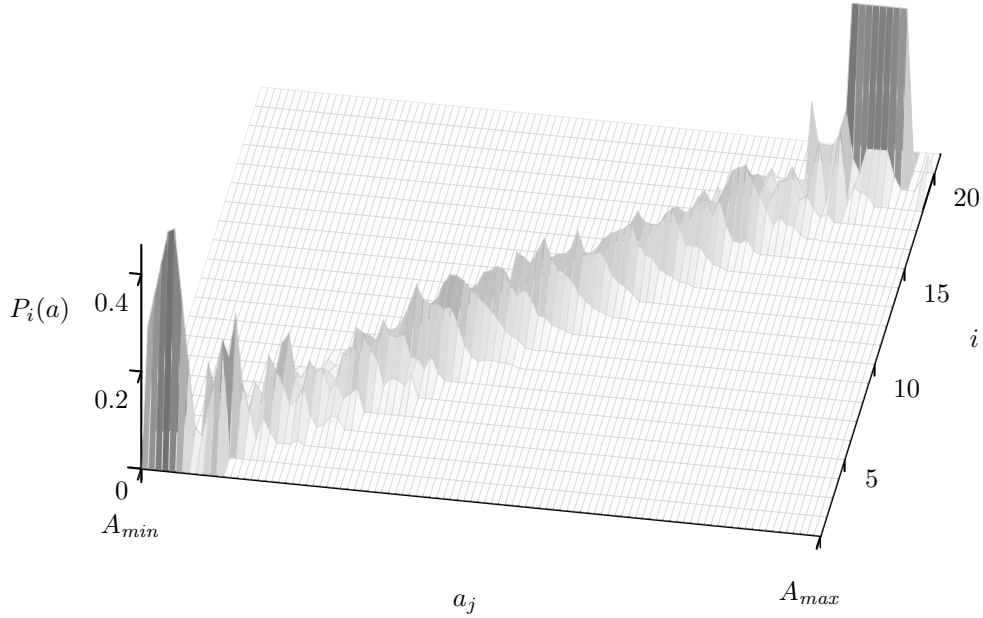


Figure 7.26: Individual transition probabilities $P_i(a)$ for all 21 quantization stages i of the first partial amplitude trajectory.

Individual transition probabilities for the sixth partial frequency trajectory of single sound item 01 are shown in Figure 7.27 as a transition matrix. The distributions are wider in general, indicating more rapid fluctuations. Detailed inspections of partial frequency trajectories in Figure 7.14(a) revealed that their variance is significantly smaller. The fluctuations thus have less impact. Figure 7.28 shows the individual transition probabilities for the energy trajectory of a single Bark band as a transition matrix. The nature of the transition probabilities is similar to those of the partial amplitudes.

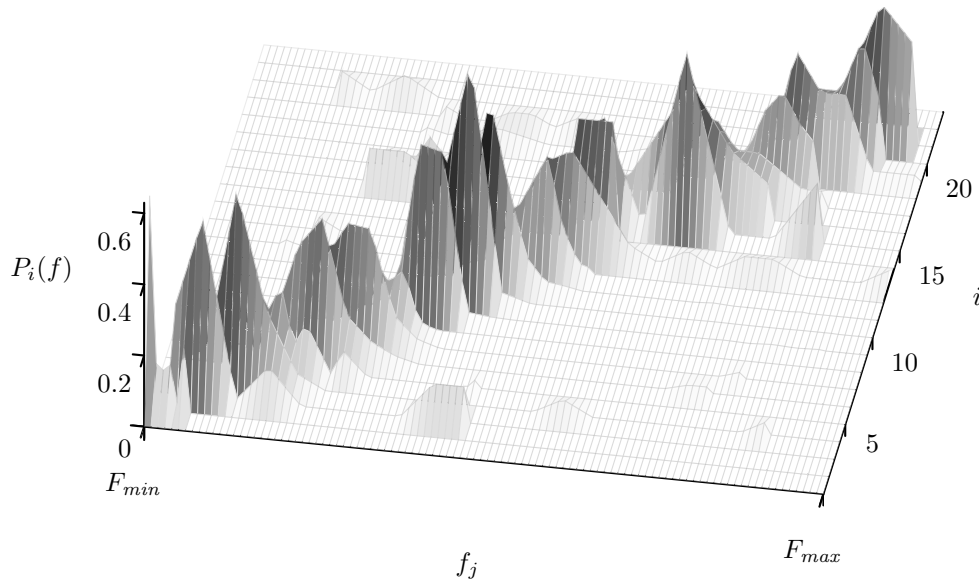


Figure 7.27: Individual transition probabilities $P_i(f)$ for all quantization stages i of the sixth partial frequency trajectory in single sound item 01.

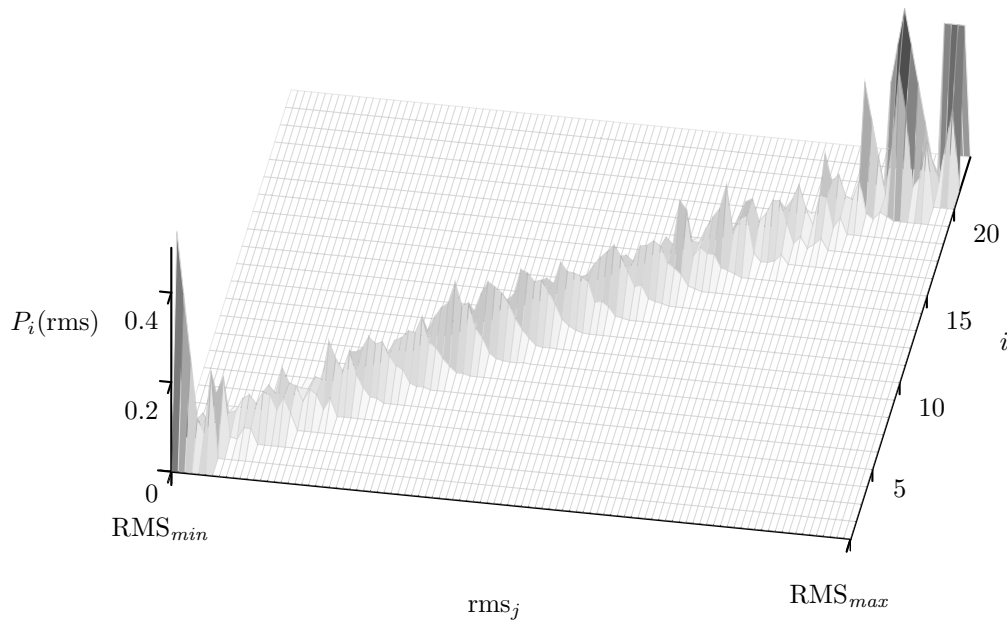


Figure 7.28: Individual transition probabilities $P_i(f)$ for all quantization stages of the fifth Bark band energy in single sound item 01.

Inverted Cumulative Mass Functions

For the application in the synthesis algorithm, inverted cumulative mass functions $ICMF_i$ have to be derived from the probability mass functions. This is performed via cumulative mass functions, in the same way as for the stateless approach, presented in Equation 7.13 and Equation 7.15.

Figure 7.29 shows the individual $ICMF$ s for the transition probabilities of the first partial's amplitude trajectory in single sound item 01. Similar to the probability mass functions, this visualization shows the frequency characteristics of the trajectory through the overlap of the sigmoidal functions. Figure 7.30 shows individual $ICMF$ s for the transition probabilities of a partial frequency trajectory and Figure 7.31 those of a Bark frequency band energy trajectory. For all examples, distributions at the boundaries have only few non-void values in their transition probabilities and an $ICMF$ with a low slope. The lower the slope and the fewer the support points of the inverted cumulative mass function, the fewer the possible transitions. Constant $ICMF$ s indicate that there is only one possible transition for the corresponding quantization stage.

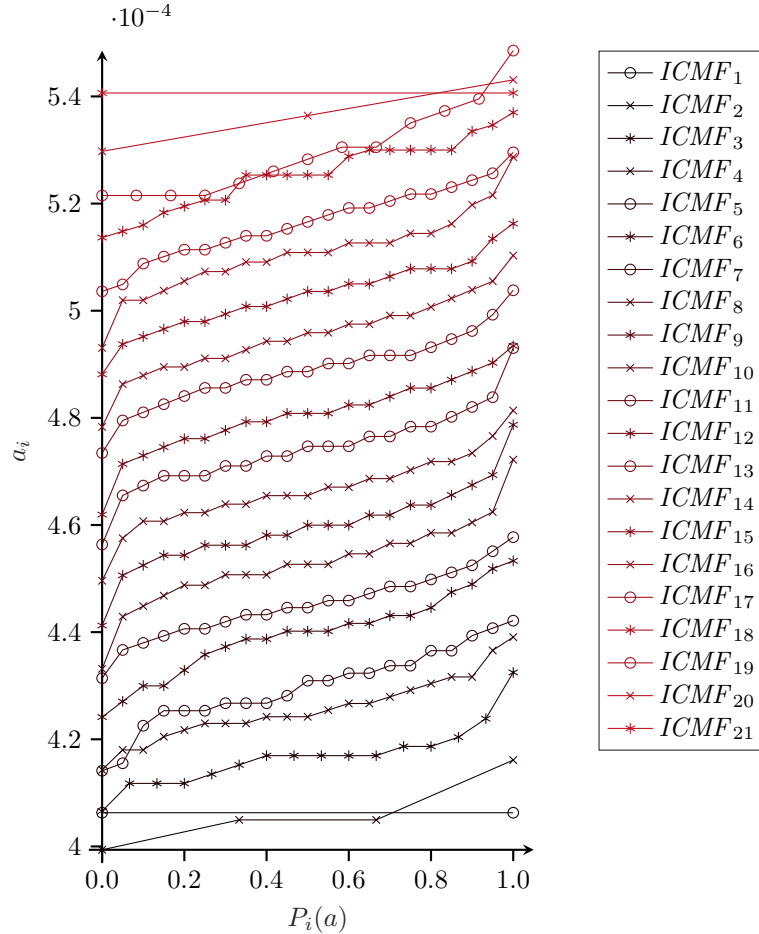


Figure 7.29: Inverted cumulative mass functions for the transition probabilities of the first partial's amplitude trajectory in single sound item 01.

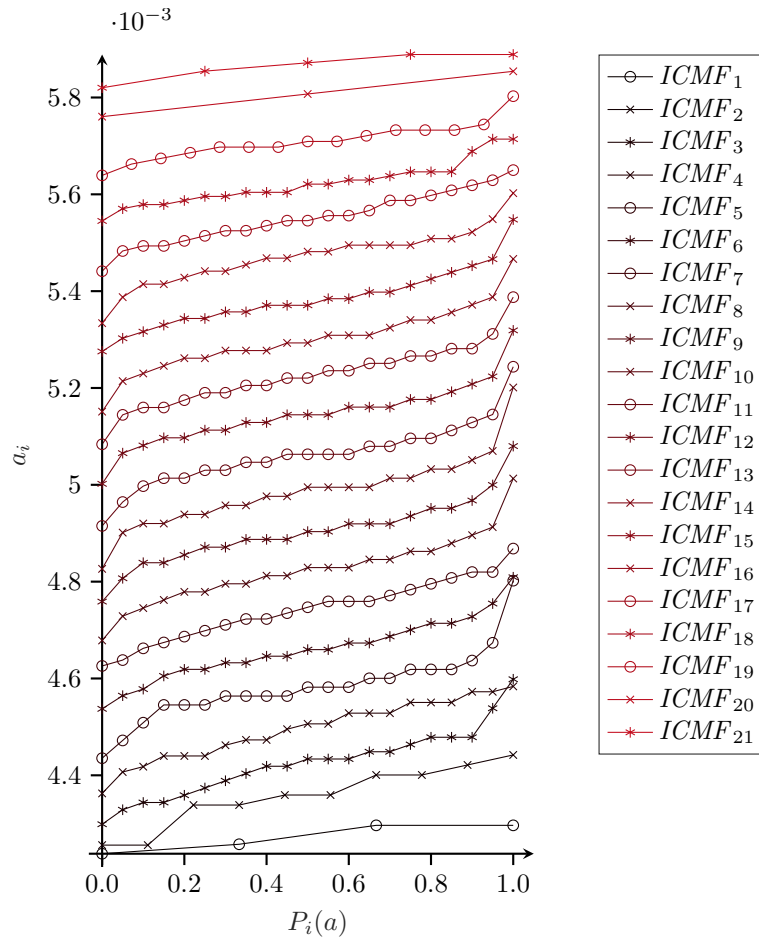


Figure 7.30: Inverted cumulative mass functions for the transition probabilities of the first partial's frequency trajectory in single sound item 01.

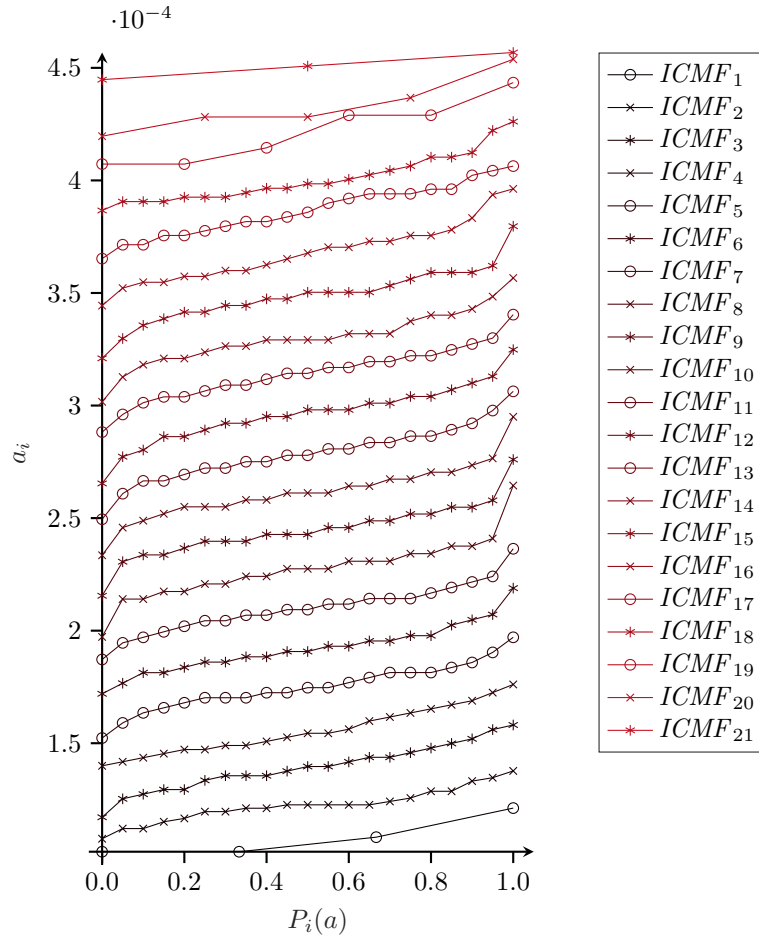


Figure 7.31: Inverted cumulative mass functions for the transition probabilities of the fifth Bark band energy trajectory in single sound item 01.

7.4 Transition Modeling

The method for statistical modeling presented in the previous Section 7.3 is only used for the sustain portions of the instrument sounds. Goal of the sustain model is to remove the time axis of the spectral modeling trajectories through the statistical model, in order to allow expressive control for excitation continuous instruments. A different approach is necessary for the remaining segments, which are accumulated in the *transition* class, according to the segmentation paradigm presented in Section 6.4. This includes *attack* and *release* segments, as well as actual transitions between two notes, in the data set captured as *legato* and *glissando*. Trajectories of the spectral modeling parameters in these segments are modeled as temporal trajectories with statistically tuned parameters.

7.4.1 Attack

The boundaries and the duration of the attack segment of each single sound are extracted from the manual annotations of the single sound part within the sample library. The synthesis algorithm allows the use of the original trajectories for the partial amplitudes, frequencies and the noise band energies within these boundaries. Since onset transients of musical instruments have a distinct idiosyncratic behavior, the use of original attack segments is common in related synthesis approaches. Additionally, linear slopes can be used to synthesize the parameter trajectories during the attack.

7.4.2 Release

Exponential Model

The temporal boundaries of the release segments are included in the annotation data of the sample library. The most important parameter to be modeled for the release segment is the partial amplitude. For modeling the decay during the release, an exponential trajectory is used, based on the number of samples N of the transition and an exponential factor λ . An additional linear scaling factor $-\frac{n}{N} + 1$ assures that the trajectory reaches 0 on the last sample:

$$a^* = \left(-\frac{n}{N} + 1\right) e^{\frac{-\lambda n}{N}} \quad (7.17)$$

Figure 7.32 shows exponential release trajectories with different values for λ . The higher λ , the steeper the release. Negative values are not expected, considering the typical release behavior.

Calculating Individual λ Values

A λ value is calculated for every partial trajectory in each item of the single sounds in the sample library. This is achieved with a step-wise optimization algorithm, minimizing the sum of the squared error e between the measured release trajectory a and the exponential model a^* :

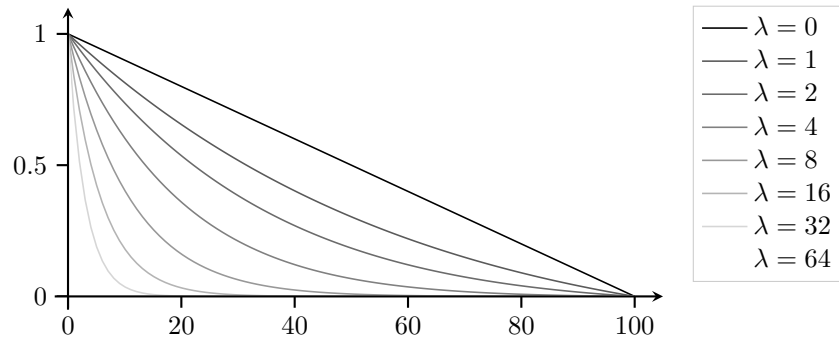


Figure 7.32: Release trajectories from the exponential model with different values for λ .

$$e = \sum_{n=1}^L (a[n] - a^*[n])^2 \quad (7.18)$$

All release segments are smoothed with a 50 sample moving average filter and normalized by their first sample. Figure 7.33 shows original and modeled release trajectories of the first four partial amplitudes from the single sound item 40.

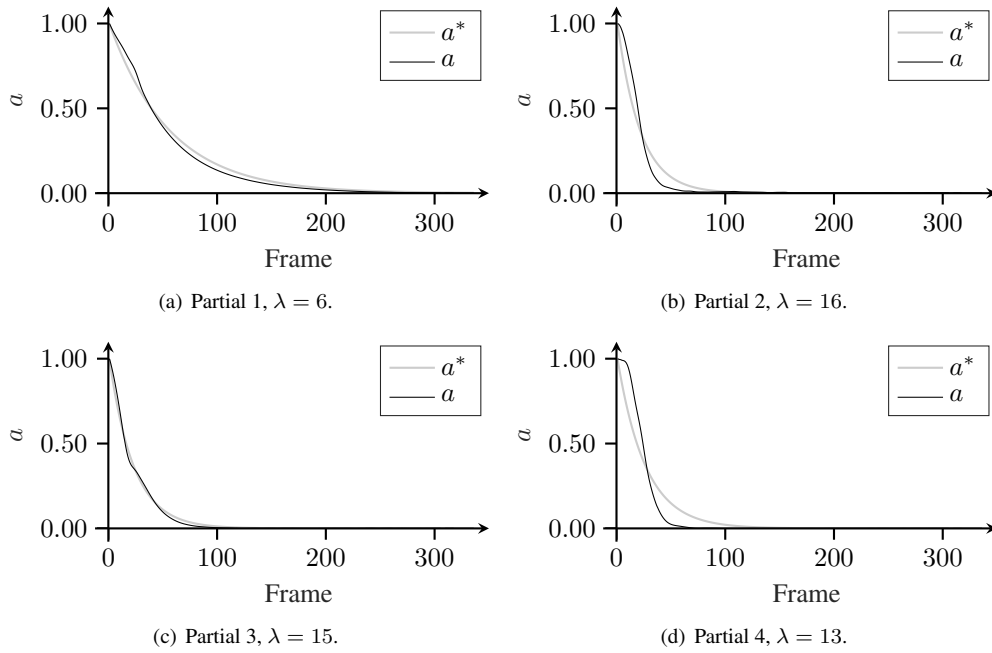


Figure 7.33: Normalized original and modeled release trajectories of the first four partials from item 40 of the single sound items.

Group Statistics

In order to model the variance in the release trajectories, group-wise statistics are extracted from the individual λ values. Using these statistical values during synthesis, each release will have a slightly different character, increasing the vividness of the approach. All sounds in the single sound category are therefore grouped into eight subsets of the timbre plane, as listed in Table 7.1.

Table 7.1: Subsets for the release modeling

MIDI Pitch	pp - mp	mf - ff
55-66	Group 1	Group 5
67-78	Group 2	Group 6
79-90	Group 3	Group 7
91-105	Group 4	Group 8

Within each group, mean and standard deviation of the exponential factor λ are obtained for every individual partial. Figure 7.34 shows the group-wise dependency of λ from the partial index, alongside a smoothed version, calculated with a moving average of five samples. All plots show a local maximum below the twentieth partial, with a decrease towards the lower indices. This encodes the slower decay for the first partials. Higher partials have steeper decay slopes and thus a more rapid decrease in energy. λ values for groups with higher fundamental frequencies are overall larger than those for lower fundamental frequencies. In addition, they show a more prominent, earlier increase of λ towards high partials, since this relation depends on the partial frequency, not on the partial index.

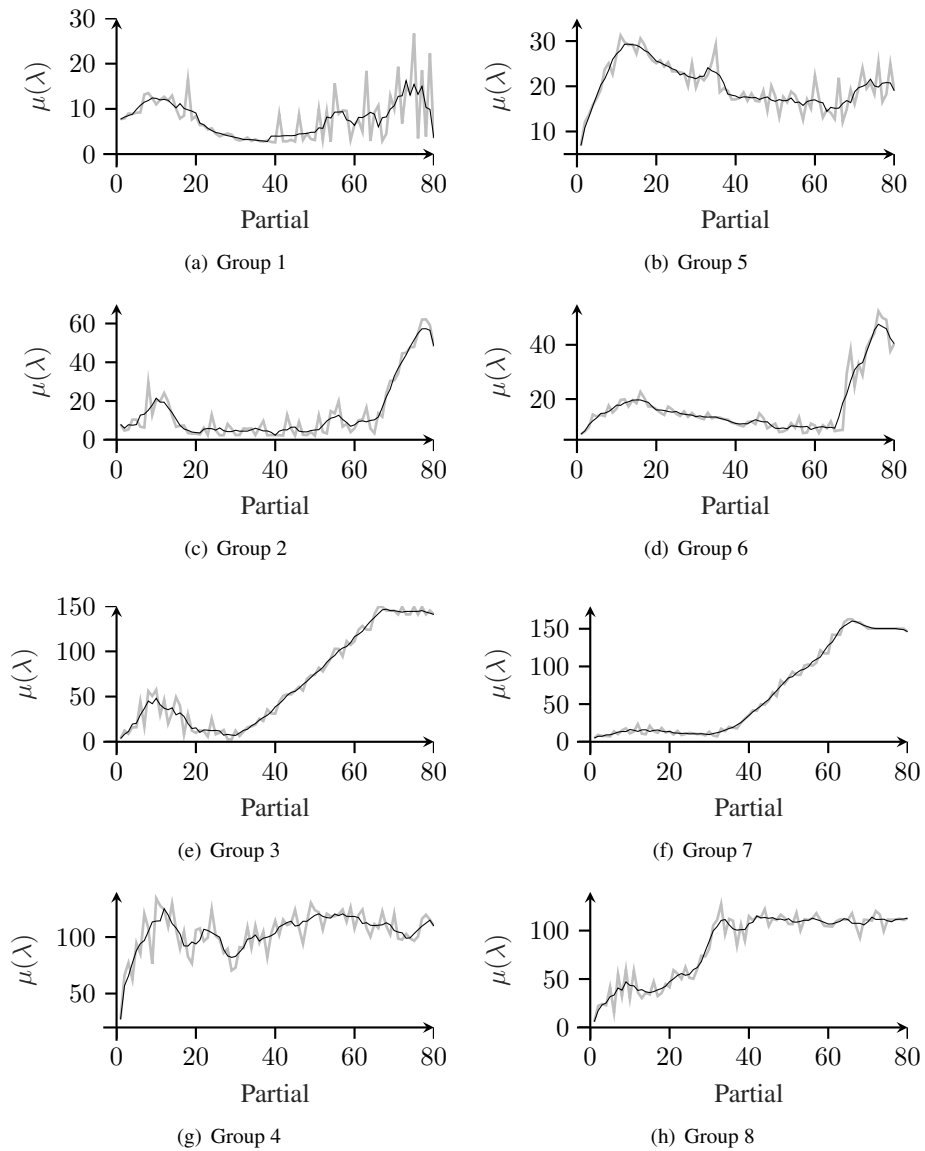


Figure 7.34: Mean λ values (gray) and smoothed version (black) for all partials, grouped according to Table 7.1.

7.4.3 Glissando

The central feature of interest for glissando transitions is the fundamental frequency and consequently the the partial frequency trajectories. As outlined in Section 4.2.4, the modeling of glissando frequency trajectories has been common in conventional synthesizers for many years. It enables glissando techniques in keyboard-based instruments, which lack a true continuous frequency control. Although the GLOOO synthesis system can be controlled with continuous frequency input and the direct shaping of frequency slopes is possible, automated glissandi are implemented for the use with key-based systems, such as the BINBONG, introduced in Chapter 9.

A previous user study investigated the interactive use of different trajectory models for glissando pitch transitions (von Coler, Götz, & Lepa, 2018). It included a hyperbolic tangent, cubic splines and Bézier curves. Participants were instructed to reproduce original glissando transitions from stimuli with these models. Parameter control was granted through rotary potentiometers. Results showed a significantly higher modeling error for the hyperbolic tangent but also indicated an easier use of this model, since it is controlled by only one parameter. In order to allow a simple integration and an intuitive control, the hyperbolic tangent is thus selected for modeling the glissando trajectories in the GLOOO system. Previously, the following formula has been proposed, based on the work of Götz (2018):

$$f_0(t) = c + d \tanh\left(\frac{t-a}{b}\right) \quad (7.19)$$

With

$$d = \frac{|f_1 - f_2|}{2} \quad (7.20)$$

$$c = \min(f_1, f_2) + d \quad (7.21)$$

and

$$a = \frac{|t_1 - t_2|}{2}. \quad (7.22)$$

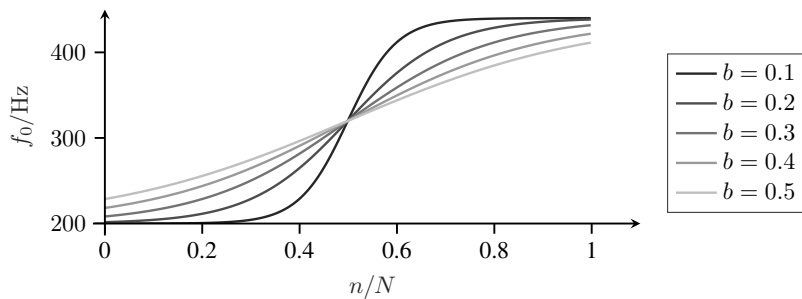


Figure 7.35: Hyperbolic tangent of the length N with different values for λ (von Coler, Götz, & Lepa, 2018).

For high values of b , the regular scaled hyperbolic tangent in Figure 7.35 does not reach the target values f_1 and f_2 . Hence, a modified version is proposed, mixing the tangent with a linear slope, to compensate this effect:

$$f[n] = (1 - \gamma^2) \left(\frac{f_2}{f_1} \frac{n}{N} + f_1 \right) + \gamma^2 \tanh \left(\operatorname{sgn}(f_2 - f_1) 10 \gamma \left(\frac{n}{N} - 0.5 \right) \right) \quad (7.23)$$

Figure 7.36 shows the extended hyperbolic tangent for different parameter settings. For values of $0 < \gamma \leq 1$ the transition blends between a linear and a sigmoidal trajectory. For values of $\gamma > 1$ the trajectory shows overshoots. As introduced in Section 2.3.1, such overshoots occur in glissando note transitions and are thus desirable in a parametric glissando model.

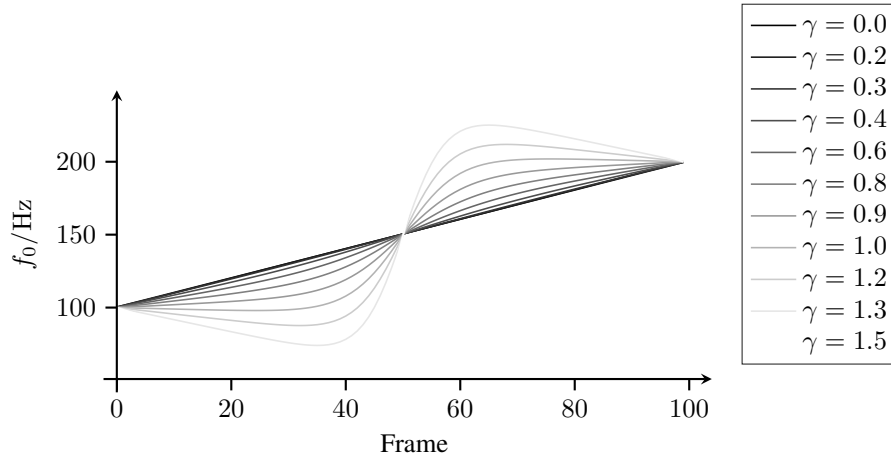


Figure 7.36: Corrected hyperbolic tangent with different values for b .

7.5 Final Data Set

For the use within the synthesis software, the results of the analysis and modeling stage are exported as a two-part data set. Different formats are used for the deterministic data, respectively the temporal trajectories of the spectral modeling and the statistical data from the Markovian modeling. Human-readable formats are chosen for both subsets, in order to increase the manageability during the development and evaluation process.

7.5.1 Deterministic Data

All trajectories resulting from the spectral modeling in Section 7.2 are stored in tab-separated text files as the *deterministic model*. They can be used for reproducing the original sound but also have a use in the statistical synthesis. Parts of the trajectories can be used for synthesizing the attack and release segments, depending on the selected synthesis mode. Deterministic data includes the fundamental trajectory for each file in a single-column file with the extension *.F0. Trajectories for partial amplitudes are stored in a *.AMPL file and partial frequencies in a file with the extension *.FREQ, both with N_{part} columns. Although not used in the synthesis at this point, the partial phases are stored in a separate *.PHA file with the same number of columns. The 24 Bark band energy trajectories are stored in a file with a *.BBE extension. For all 336 items in the single sound data set this results in

$$N_{det} = 336 \cdot 5 = 1680 \quad (7.24)$$

files for the deterministic part. A memory of 2.7 GB is needed for storing this part of the model in scientific notation with a precision of six decimal digits. Keeping the deterministic data in this format is beneficial during the development stage but takes a considerable amount of memory and a long time for loading the whole set.

7.5.2 Statistical Data

The *statistical model* is exported in the YAML format (Ben-Kiki, Evans, & Net, 2020). YAML has the advantage of being structured and human-readable and is thus well suited for the development process. Furthermore, import and export options are given in most programming languages. This ensures a flexible export from the Matlab analysis software, as well as a smooth import into the C++ synthesis engine. Single aspects of the data set can be visualized at any time. However, in the current implementation the import is significantly slower than with binary formats or plain text data and the use of an alternate format may be necessary at a given time.

The statistical data of each item in the single sound part of the sample library is exported as an individual YAML file, structured as shown in Figure 7.37. Individual sounds can be synthesized from each single file, whereas the complete set is required for the full expressive synthesis.

The root node `param` contains all analysis parameters, some of which are helpful for debugging and traceability, whereas others are relevant to the synthesis process. This includes the sampling rate of the audio files, the hop size of the partial tracking and the Markovian modeling parameters. The root node `INF` holds the basic information on the item, such as the fundamental frequency and dynamic level. Parameters for attack, sustain and release are stored in separate root nodes.

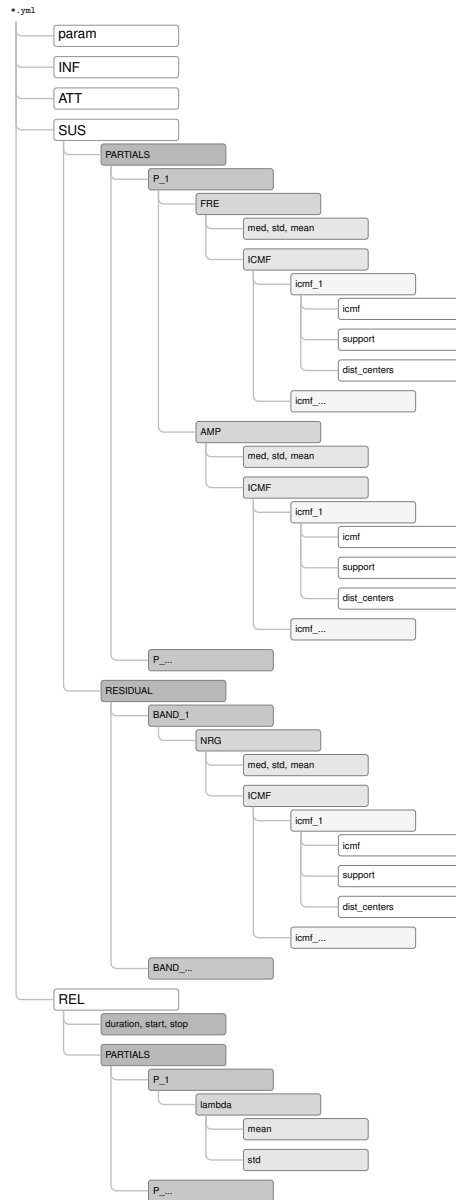


Figure 7.37: YAML data structure for a single item in the stochastic data set.

The sustain node `SUS` contains two sub-nodes, one for the partials and one for the residual component. Within the `PARTIALS` node, each partial is represented as an individual sub-node, containing statistical data for partial amplitude in the node `AMP` and partial frequency in the node `FRE`. For both parameters, statistical data is comprised of nodes with basic distribution features, including `mean`, `std` and `median`, and a node `ICMF`, holding the data for the statistical modeling synthesis, more precisely the `ICMF` itself, as well as the support points and the distribution centers. The `RESIDUAL` node contains a sub-node for the energy model of each Bark band. As for the partial parameters, the basic statistical properties are stored, alongside the statistical model based on the `ICMF`.

The `REL` node holds one sub-node for basic temporal properties, such as length and start/stop of the segment within the deterministic data trajectories. Within the sub-node `PARTIALS`, mean and standard deviation are stored for the λ value of each individual partial in dedicated nodes.

Synthesis System

This chapter describes the the GLOOO system for spectro-spatial sound synthesis. Section 8.1 is dedicated to the underlying theory and algorithmic principles of statistical spectral synthesis, developed in the course of this project. This includes details on the novel Markovian Inverse Transform Sampling Synthesis. The concept of spatial synthesis, respectively the theory of spectral dispersion is introduced in Section 8.2. Section 8.3 explains the implementation of the algorithms in the real-time C++ synthesis software in detail. Finally, relevant properties of the statistical spectral synthesis are presented in Section 8.4 through measurements of the synthesis software.

8.1 Statistical Spectral Synthesis

The analysis and modeling stage presented in Chapter 7 results in a data model, which is the basis for the synthesis system introduced in this section. Due to the nature of this model, the system allows different synthesis modes. One of them is referred to as the *deterministic mode*, which makes direct use of the feature trajectories obtained through conventional spectral modeling. Since this is an established, well documented process (McAulay & Quatieri, 1986; Serra & Smith, 1990; Levine & Smith, 1998), it will not be treated in this thesis. However, the different modes of statistical synthesis for the sustain part have been developed in the course this project and are thus introduced on the following pages. They include the *Mean Mode*, the *Stateless Inverse Transform Sampling* and the *Markovian Inverse Transform Sampling*. Initially, the general concept of *timbre-space interpolation* will be introduced, which is relevant for all methods of statistical synthesis.

8.1.1 Timbre-Space Interpolation

The single sound part of the data set samples the instrument within the timbre plane. A grid of 336 support points with different combinations of pitch and intensity, explained in detail in

Section 6.2.1, is used for this purpose. Without further measures this data allows the synthesis of sounds with characteristics of the samples at these exact support points. But when applying this discrete model in expressive sound synthesis, it is necessary to also synthesize any arbitrary point in the timbre plane. As stated by Bonada and Serra (2007) in a similar context, expressive synthesis systems should be able to create any trajectory within the sampled space.

An extension of the data set beyond the sampling grid is realized with an interpolation technique. Similar to the interpolation approaches presented by Bowler, Purvis, Manning, and Bailey (1990) and Rován et al. (1997), timbre space interpolation extends the sample library for the use with continuous control signals. Bowler et al. (1990) use the term *performance parameters* for the control parameters of the DMI. The performance parameters span an orthogonal space. During synthesis, the *articulation vector* is moved through this space to allow an expressive performance. The space is discrete, whereas the articulation vector is moved continuously. Thus, an interpolation between the nearest support points is necessary. Bowler et al. (1990) propose an interpolation method based on simplex polytopes, resulting in first-order-continuous interpolation functions. Rován et al. (1997) list several shortcomings in their approach, mainly caused by the strongly limited number of nine sampling points in the timbre plane. Using the more densely sampled TU-Note Violin Sample Library, combined with the statistical approach introduced in Sections 8.1.3 and 8.1.4, these issues are resolved.

The interpolation technique proposed in the GLOOO synthesis system is based on distance measures to the nearest samples in the data set. Figure 8.1 shows a point P with arbitrary pitch and intensity, alongside the distance vectors to the four nearest support points A , B , C and D .

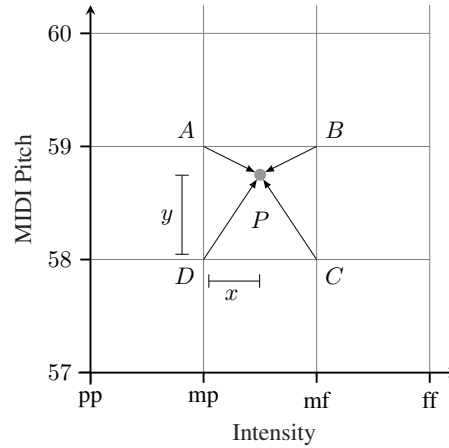


Figure 8.1: Support points A , B , C and D with an arbitrary point P in the timbre plane.

Depending on the synthesis modes presented in the following section, different steps are necessary to allow a smooth, interpolated synthesis. Yet, the same distance measures are used for each mode to generate the weights $w_{A,B,C,D}$ for each neighboring support point. Each weight is calculated as a variation of the product of the lengths of the normalized projections x and y on the two dimensions pitch and intensity. The projections are normalized such that $x = 1$ and $y = 1$ for $P = B$. In this way, the sum of weights always results in 1:

$$w_A = (1 - x)(1 - y) \quad (8.1)$$

$$w_B = x(1 - y) \quad (8.2)$$

$$w_C = xy \quad (8.3)$$

$$w_D = (1 - x)y \quad (8.4)$$

8.1.2 Mean Mode

The mean mode is the most basic way of using the statistical data model for synthesis. It makes direct use of either the mean or the median values stored in the data set for each spectral modeling parameter. For the sake of simplicity, it is referred to and explained as the mean mode. In each synthesis frame, the synthesis algorithm processes the latest control values for pitch and intensity. The four nearest support points are found and the mean values μ_i are obtained for all spectral modeling parameters, more precisely for all partial amplitudes and frequencies and for all Bark band energies of the related samples. In this scenario, the weights can be directly used to generate the interpolated values of all spectral synthesis features:

$$\tilde{\mu} = w_A \mu_A + w_B \mu_B + w_C \mu_C + w_D \mu_D \quad (8.5)$$

Since the mean mode always generates the exact same spectral modeling parameters for a given combination of pitch and intensity, the resulting sounds do not contain the previously emphasized jitter or shimmer, caused by stochastic signal components. If the control parameters remain constant, the synthetic sounds remain perfectly stationary, whereby they lose the vividness of original instrument sounds. However, when used in combination with an expressive interface or driven by dynamic control trajectories, this drawback is eliminated to some extent, since the thus induced fluctuations in the control parameters pitch and intensity inherently cause fluctuations in timbre. Compared to the following statistical approaches, the mean mode is computationally less expensive and requires only a small part of the data in the complete model, since the ICMFs are not needed. It is thus an interesting option for a lightweight implementation on limited systems.

8.1.3 Stateless Inverse Transform Sampling Synthesis

In contrast to the mean method, inverse transform sampling synthesis is able to reproduce the so called *harmonic residuals* (Lindemann, 2010), the rapid fluctuations in partial parameters which are not directly controlled by the musician and contribute to the timbre of the instrument. The method of *Stateless Inverse Transform Sampling Synthesis* constitutes a simplified version of the later introduced Markovian approach. Although the recent version of the synthesis software uses only the Markovian approach, a description of the simpler stateless algorithm for statistical spectral synthesis is useful, since it introduces the basics for the extended Markovian version and can still be considered for upcoming implementations.

The stateless approach makes use of the stateless statistical modeling results in the data set, presented in Section 7.3.2. At this stage, each parameter of the spectral model is represented by a single distribution function, describing its statistical properties. During synthesis, inverse transform sampling (Devroye, 1986, p. 85) is used to generate stochastic parameter trajectories. This process relies on the inverted cumulative mass function ICMF, as shown in Figure 7.21. The ICMF can be interpreted as a mapping function between a uniform distribution $\mathcal{U}(0, 1)$ and the target distribution $P(X \leq x)$. Thus, a pseudo-random number generator can be used to create the desired distribution via inverse transform sampling.

Inverse transform sampling can be implemented using a sequential search method (Devroye, 1986, p. 85) on the cumulative mass function, without actually inverting the distribution functions in advance. For a random value $0 \leq r \leq 1$ from the uniform distribution, the corresponding value \tilde{r} from the target distribution for the variable x can be obtained as the argument of the minimum of the difference to the relevant cumulative mass function:

$$r^* = \arg \min_r [\text{CMF}_x - r] \quad (8.6)$$

In initial stages of the implementation in this project, this was realized using a vector search from the C++ Standard Library for Equation 8.6. Binary search trees can increase the efficiency of this approach and lookup tables or *guide tables* for the individual distributions are even more efficient (Devroye, 1986). For the amount of parameters in the synthesis engine, the sequential search showed to be efficient enough. Newer versions of the synthesis algorithm rely on an inverse transform sampling implementation based on the actual ICMFs, alongside the associated vector of support points. Both are stored within the data set for all spectral modeling parameters. In this case, the generated pseudo-random numbers can be directly mapped to the values of the ICMF vector by scaling them to the vector size. The process of inverse transform sampling from a distribution ICMF with the random value r is in the remainder denoted as

$$r^* = \mathcal{I}(\text{ICMF}, r). \quad (8.7)$$

Inverse transform sampling can be used to generate interpolated random values \tilde{r}^* for any combination of pitch and intensity located within the limits of the support points of the sample library. This is achieved by a weighted sum of dependent stochastic values from four inverse transform sampling processes, using the same uniform random number r :

$$\begin{aligned} \tilde{r}^* = & w_A \mathcal{I}(\text{ICMF}_A, r) + w_B \mathcal{I}(\text{ICMF}_B, r) \\ & + w_C \mathcal{I}(\text{ICMF}_C, r) + w_D \mathcal{I}(\text{ICMF}_D, r) \end{aligned} \quad (8.8)$$

The proposed algorithm is shown as a flowchart in Figure 8.2. Initially, a single random value r is generated from a uniform distribution. This value is used in inverse transform sampling for each of the four neighboring support points to create the random values $\tilde{r}_{A,B,C,D}$. Subsequently, the related weights $w_{A,B,C,D}$ are applied. Since inverse transform sampling has been performed with the same r , the four resulting values are statistically dependent and can be summed up to

get the interpolated random value \tilde{r}^* at point P .

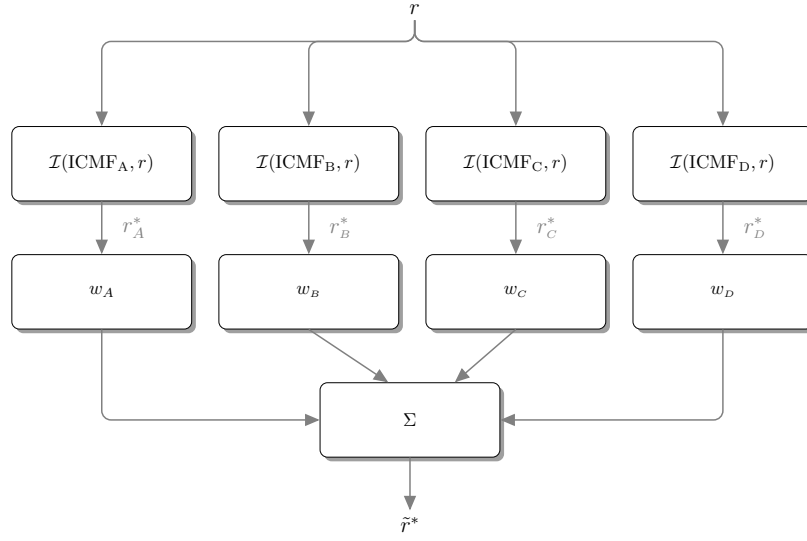


Figure 8.2: Interpolated inverse transform sampling.

Since the above presented algorithm does not model the temporal behavior of the stochastic processes, the resulting trajectories have different spectral features than the originals. More precisely, undesired jumps occur, since consecutive random values are independent. These jumps can be smoothed out through low-pass filtering. A sliding window moving average with 10 values has been used for this in early versions of the synthesis system, based on a circular buffer. However, this manipulation affects the stochastic process by narrowing the original distribution. This distortion can be counteracted by widening the distributions prior to filtering, but ultimately this problem led to the development of the Markovian approach. The stateless inverse transform synthesis has been excluded from the development branch of the GLOOO software in order to reduce model size and complexity. However, it may be of interest for following investigations.

8.1.4 Markovian Inverse Transform Sampling Synthesis

Markovian Inverse Transform Sampling Synthesis (*MITSSy*) combines the inverse transform sampling algorithm introduced in the preceding Section 8.1.3 with the principles of Markov processes. In this way, spectral properties of the analyzed trajectories can be preserved with probabilistic means. For that reason, the statistical data set includes the transition probabilities in the form of inverse cumulative mass functions, as shown in Figure 7.29. These sets of transition probabilities for a single parameter of one sound in the data set are further referred to as ICMF collections.

A flow chart for the *MITSSy* algorithm is presented in Figure 8.3. Essentially, it extends the core principles of the stateless inverse transform sampling algorithm with a feedback and uses the transition probabilities in the synthesis data set, rather than single distributions for each spectral modeling parameter.

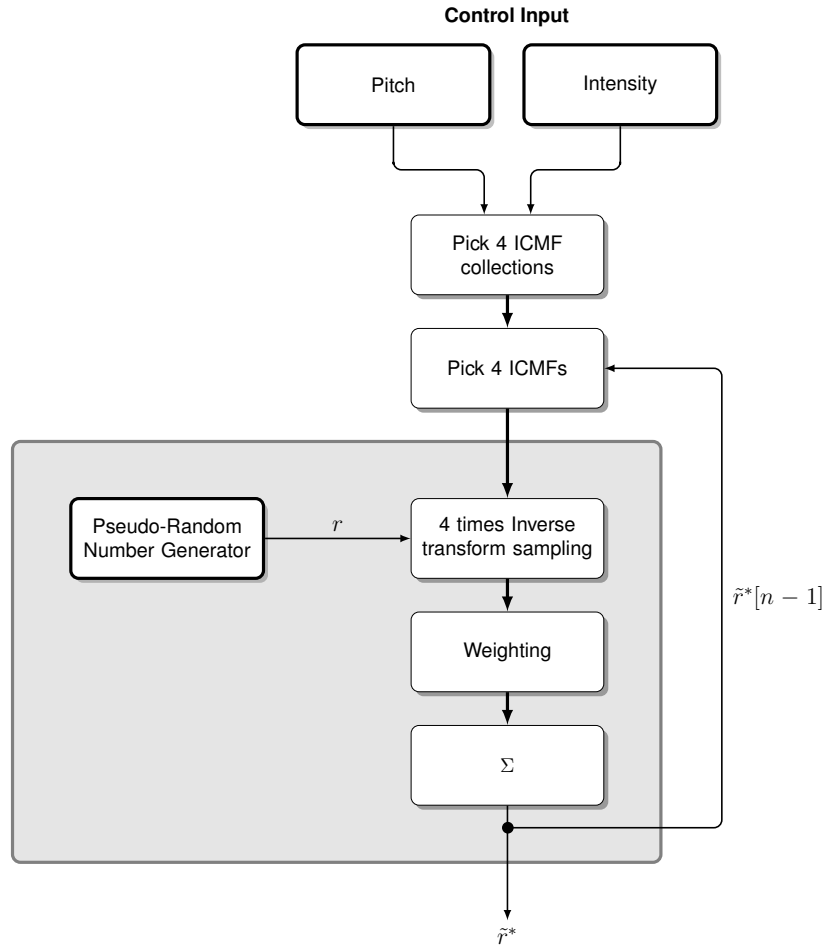


Figure 8.3: Flow chart for the Markovian Inverse Transform Sampling Synthesis for a single parameter with the algorithm for interpolated inverse transform sampling from Figure 8.2 inside the gray box.

The process is fed with the continuous control parameters pitch and intensity. According to the interpolation principle, *ICMF* collections of the four nearest samples are picked from the data set for each synthesis frame, based on these control parameters. Within each collection, the relevant *ICMF* is picked according to the past stochastic value $\tilde{r}^*[n-1]$ from the feedback loop. Inverse transform sampling is then performed for each of the four distributions, as illustrated in Figure 8.2, using a single random number from the uniform distribution.

8.2 Spectral Dispersion

The GLOOO synthesis algorithm is specifically designed for the use with external spatial rendering software, to enable spectro-spatial sound synthesis. For this reason, all harmonic and residual components of the spectral model can be routed to the outputs of the software, individually. This includes up to 80 partials and the 24 noise signals from the filter bank.

To this point, spatialization in this system is performed in an object-based or point-source fashion. Thus, all spectral components can be assigned to a point sound source with a virtual position. *Spectral Dispersion* refers to the mapping of spectral components to point sources in the spatialization stage. In terms of the synthesis engine itself this is equivalent to the mapping of spectral components to the individual JACK outputs of the synthesizer. The kind of dispersion has a significant influence on the interaction of spectral and spatial attributes. Three different dispersion modes are implemented in the synthesis engine and will be introduced in the following paragraphs.

8.2.1 Uniform Dispersion

The uniform dispersion represents the partial- and frequency-independent case, minimizing the entropy in the spectral distribution. Figure 8.4 visualizes the concept for connecting 80 partials P_n and 24 noise bands N_n to and M sources S_m . All spectral components are summed and the resulting signal, containing the full synthesized sound, is routed to all outputs of the software and, in consequence, to all point sound sources. Due to this redundancy, the number of point sound sources can be freely chosen. The global gain has to be scaled by the inverse number of sources. This highly minimized dispersion mode is primarily implemented as an anchor for future experiments which aim at a perceptual evaluation of spectro-spatial sound synthesis.

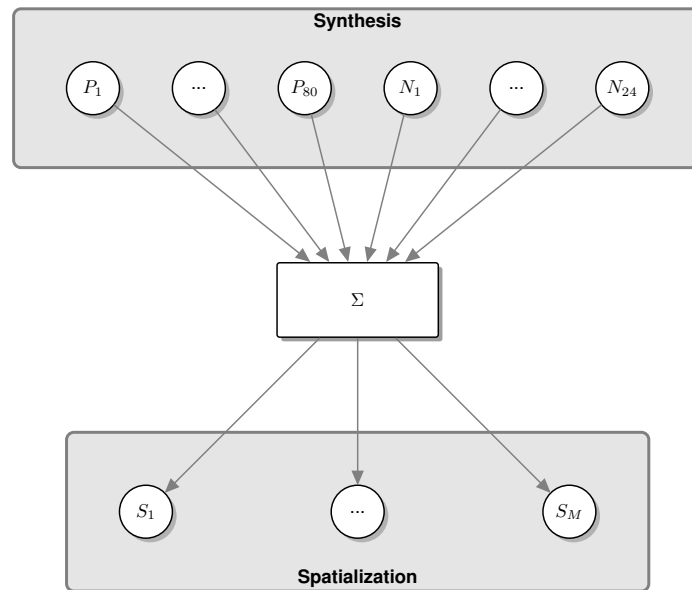


Figure 8.4: Uniform mapping of 80 partials and 24 noise bands to M sound sources.

8.2.2 Discrete Dispersion

In the *discrete dispersion* mode, each spectral component is directly sent to an individual output of the synthesis software and can thus be linked to one specific point sound source. Ideally, all partials and all noise bands have a dedicated sound source, allowing the individual spatialization of all 80 harmonic and all 24 noise components in one-to-one connections. Figure 8.5 shows the full discrete dispersion of all partials and noise bands to 104 point sound sources, as it is implemented in the GLOOO synthesizer in the recent version. For this purpose, the GLOOO synthesizer offers 104 individual JACK outputs, named after the spectral components. In order to reduce the processing load, the number of point sources can be decreased, leading to multiple spectral components sharing one source. This was necessary in earlier stages of the project, especially to relieve the spatial rendering software which caused glitches for high numbers of moving sound sources.

Due to the individual, frequency independent treatment of the partials, the discrete dispersion goes beyond the capabilities of spectral spatialization and allows true spatial sound synthesis. It is thus the default mode of the synthesis software. Due to the decreasing partial amplitude towards higher partials, point sources associated with a high partial index carry only little energy. This needs to be considered when designing mappings for the spatial arrangement of the sound sources.

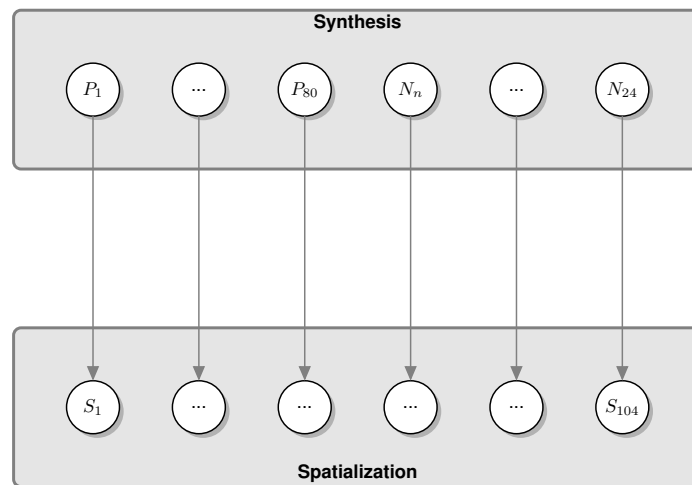


Figure 8.5: The discrete dispersion mode with 80 partials, 24 noise band signals and 104 point sound sources.

8.2.3 Filter Bank Dispersion

The *filter bank dispersion* mode distributes the spectral modeling components to a limited number of outputs, each corresponding to a frequency band. Figure 8.6 illustrates this procedure for a Bark scale frequency bank with 24 bands and a spatialization with 24 point sound sources. All of these point sound sources are fed from an individual band pass filter with the amplitude response $H_i(f)$ of one frequency band. Each of the 80 partials from the spectral synthesis is connected to all band pass filters. The level of the partials in any of the point sound sources thus depends

on their instantaneous frequency and the underlying amplitude response of the filter. The noise signals N can be directly connected to the point sources of the same index, since they have been generated with the exact same filter bank.

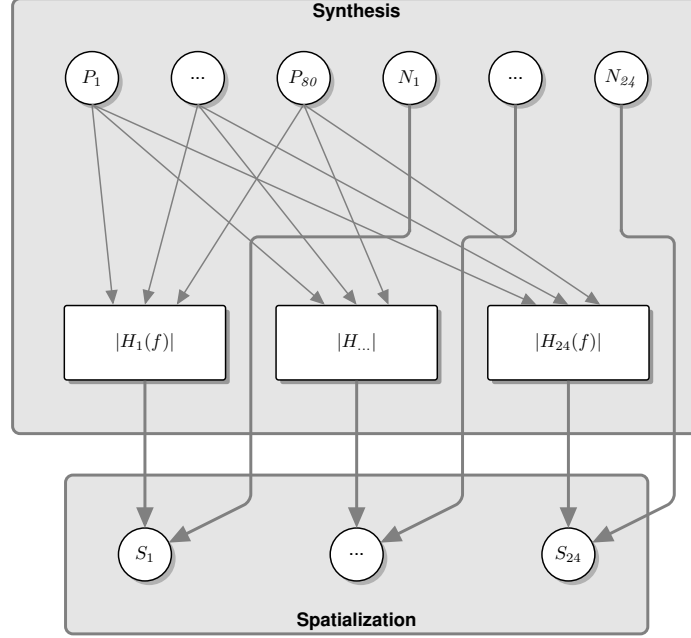


Figure 8.6: Filter bank dispersion with 24 Bark scale filters.

Spectral synthesis with filter bank dispersion has an effect similar to standard spectral spatialization of sound through filter banks, as introduced in Section 3.5.2. In contrast to the discrete dispersion, the connection between partials and virtual sound sources is dynamic in this case. Since the individual partial frequencies depend on the control parameters pitch and intensity, the spectral distribution in space is fluctuating, even if the point sound sources remain stationary. Depending on the pitch of the synthesized sounds, point sources related to low frequency bands may remain silent. Filter bank dispersion can be used for synthesizing radiation patterns of musical instruments with more flexibility than mere spectral spatialization.

Gaussian Filter Modeling

The filter bank dispersion is based on the Bark scale filter bank used in Section 7.2.3 for the modeling of the residual component. Therefore, the noise component of the statistical spectral synthesis needs no further processing and can be directly routed to the related point sources. For processing the single partials, an efficient algorithm is proposed, which approximates the amplitude responses of the single band pass filters in the filter bank with Gaussian distributions (Guo, 2011). Normalized to an amplitude of 1, each filter is expressed by the mean μ_j and the standard deviation σ_j . Since the partials consist of only one single isolated frequency f_i , their amplitude a_i on each frequency band j can be calculated with the according Gaussian transfer function:

$$a_i(j) = e^{-\left(\frac{f_i - \mu_j}{2\sigma_j}\right)^2} \quad (8.9)$$

The amplitude responses of the resulting Gaussian filters are shown in Figure 8.7. Table 8.1 lists the individual parameters of all 24 Gaussian distributions.

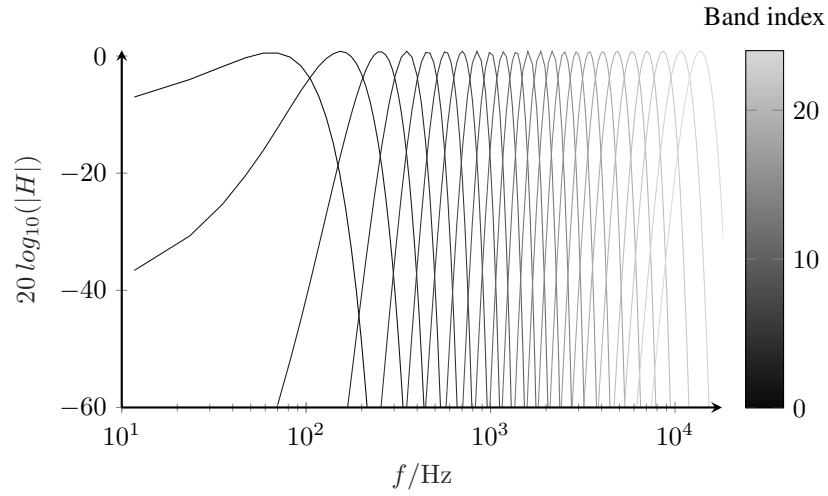


Figure 8.7: Gaussian approximations of the Bark band filter bank.

Table 8.1: Parameters of the Gaussian filter approximations.

Band Nr.	μ	σ	Band Nr.	μ	σ
1	64.74	56.56	2	154.00	68.57
3	252.43	69.20	4	351.67	69.51
5	456.65	76.41	6	571.52	83.44
7	701.70	97.34	8	846.65	103.34
9	1001.19	111.50	10	1176.94	132.31
11	1377.03	146.06	12	1601.34	168.64
13	1864.25	193.60	14	2162.93	221.26
15	2512.97	264.48	16	2928.98	313.10
17	3430.00	382.49	18	4056.67	486.54
19	4858.75	625.11	20	5860.00	763.49
21	7060.14	901.65	22	8612.18	1244.75
23	10758.04	1719.19	24	13728.25	2378.65

8.3 Implementation

The GLOOO real-time synthesis system is programmed in C++ as a free, open source project. To this day, the code was written by the author of this thesis and two students at TU Berlin. Based on a previous version and early descriptions of the algorithms, Benjamin Wiemann lay the foundation for the recent version of the GLOOO synthesizer within his master’s thesis (Wiemann, 2017). Yrkkö Äkkijyrkkä improved aspects of the software in the course of the user study in early 2020.

Specific aspects of the software are documented in the customary places in detail. Information on compilation and basic operation can be found in the project’s *Git* repository (von Coler, 2020c). A thorough documentation of the source code, created with *Doxygen*, is available at the project’s main web site (von Coler, 2020d). This section of the thesis aims at explaining the program structure with a more detailed inspection of the implementation of the core algorithms for statistical synthesis. Furthermore, all parameters and configuration options will be listed and explained.

8.3.1 Dependencies and Runtime Environment

The implementation is based on the *JACK* API for audio connectivity, with the respective C++ wrappers. This allows the use of the synthesizer on professional Linux audio systems for research and music production, in combination with other JACK-capable software. The GLOOO synthesizer implements an *Open Sound Control* (OSC) server for all control commands, realized with the *LibLO* library. Configuration files and the data for the statistical models are managed in YAML files with use of the *yaml-cpp* library. All required libraries are listed in Table 8.2.

Earlier versions of the software also included the *libsndfile* (de Castro Lopo, Erik, 2020) for reading and writing WAV-files, the *libfftw3* (Frigo & Johnson, 2020) for the Fourier transform in the initial IFFT approach, as well as the *jackcpp* library (Norman, Alex, 2012) and the *Boost* library (Dawes, Beman and Abrahams, David, 2020).

Table 8.2: Libraries installed on the development system

Library	Reference	Purpose
jack2	Davies, Paul, 2020	JACK audio API.
liblo	Harris and Sinclair, 2020	OSC support.
yaml	Ben-Kiki, Evans, and Net, 2020	Data set and configuration files.

The synthesis system is designed to compile and run on Linux systems with the necessary libraries installed. *CMake* (Kitware Inc., 2020) is used for the compilation and installation process. A *Qt Creator* project is also provided for debugging and development. So far, the software has only been used on Debian based systems with an x86_64 architecture. However, the nature of the involved tools offers the prospect of porting it to most common platforms in the future.

8.3.2 GLOOO Classes and Structure

Figure 8.8 shows a partial class diagram of the GLOOO synthesis system. It includes only those classes, attributes and methods which are relevant for describing the implementation of the algorithms for statistical spectral synthesis and the basic structure of the program. These aspects will be introduced in the remainder of this section. A detailed representation of the structure with inheritance diagrams and implementation details is included in the source code documentation (von Coler, 2020d).

GloooSynth

GloooSynth is the central class, initialized by the main function. It owns all major classes, alongside a global math object. It provides the JACK client with the buffer `**out` for the audio output and implements the callback function `process()`, called by the JACK server for every new audio output buffer.

GloooConfig

The class GloooConfig reads the YAML configuration file and implements getter methods for making the values available. This includes global parameters, like paths for the location of the data set and OSC ports, which are not changed during runtime. All configuration parameters are listed and explained in Section 8.3.5.

DeterministicData

The class DeterministicData holds a vector of 246 DeterministicSound objects and manages reading their data from the tab-separated text files. The full number of 336 items in the sample library is reduced by eliminating all multiple pitches. DeterministicData implements the method `get_trajectory_value()` for accessing the values of all spectral parameters at any point in the trajectory for all 246 items.

DeterministicSound

The class DeterministicSound holds matrices for the temporal trajectories of the parameters frequency and amplitude of all 80 partials, as well as the temporal trajectories of all 24 bark band energies for a single sample. The method `get_trajectory_value()` grants access to the trajectory values.

StochasticData

The StochasticData class manages all 246 single StochasticSound objects. As for the deterministic data set, not all 336 samples are loaded since multiple pitches are discarded. This class is responsible for reading the individual stochastic models from YAML files and implements

the algorithm for timbre-space interpolation between stochastic values from multiple individual sounds in the method `get_stochastic_value()`.

StochasticSound

Each `StochasticSound` object holds the statistical distributions and additional parameters for all 80 partials and all 24 noise bands of one item in the single sound part of the sample library. The class implements the method `get_stochastic_parameter()` for generating stochastic values from these distributions.

StatisticalParameter

`StatisticalParameter` reads the statistical model for a single spectral modeling parameter from a YAML file. An instance is used for every partial amplitude and frequency, as well as for the noise band energy of every item in the data set. The class manages access to the stored data vectors `support_points`, `icfm_collection`, `xval_collection` and `dist_centers`, with the method `get_value()`.

ExponentialDecay

`ExponentialDecay` is derived from the basic trajectory class and implements the exponential model for the release segments of the partial amplitudes. Each partial has an individual release model.

VoiceManager

The `VoiceManager` class processes control data received by the `OscManager`. It holds the instances of all single voices, managing their states in the method `update_voices()`, by activating and deactivating them according to the control signals. During the synthesis process it wraps all voices in the method `getNextFrame_TD()` for summing their individual outputs and writing them to the JACK output buffer.

SingleVoice

The `SingleVoice` class holds the actual synthesis elements, more precisely an array of 80 instances of the class `Sinusoid` and a `ResidualSynth` object. Each voice has an individual fundamental frequency and intensity. The method `cycle_start_stochastic()` processes the control parameters for the voice and adjusts the synthesis parameters. Each single voice is a state machine, managing the different segment types of the voice. Three methods, `get_next_block_bark()`, `get_next_block_uniform()` and `get_next_block_discrete()` implement the different dispersion modes and generate the actual output samples of the voices.

Sinusoid

The `Sinusoid` class is a state machine for a single partial, managing all relevant instantaneous partial parameters. The method `getNextSample()` advances the state machine by one sample, regarding the instantaneous frequency, amplitude and the system's sample rate.

ResidualSynth

The class `ResidualSynth` connects the single voices with the stochastic data and generates the energy trajectories for the bark bands in the residual synthesis. This happens inside the methods `prepare_frame_stochastic()` and `get_next_value()`.

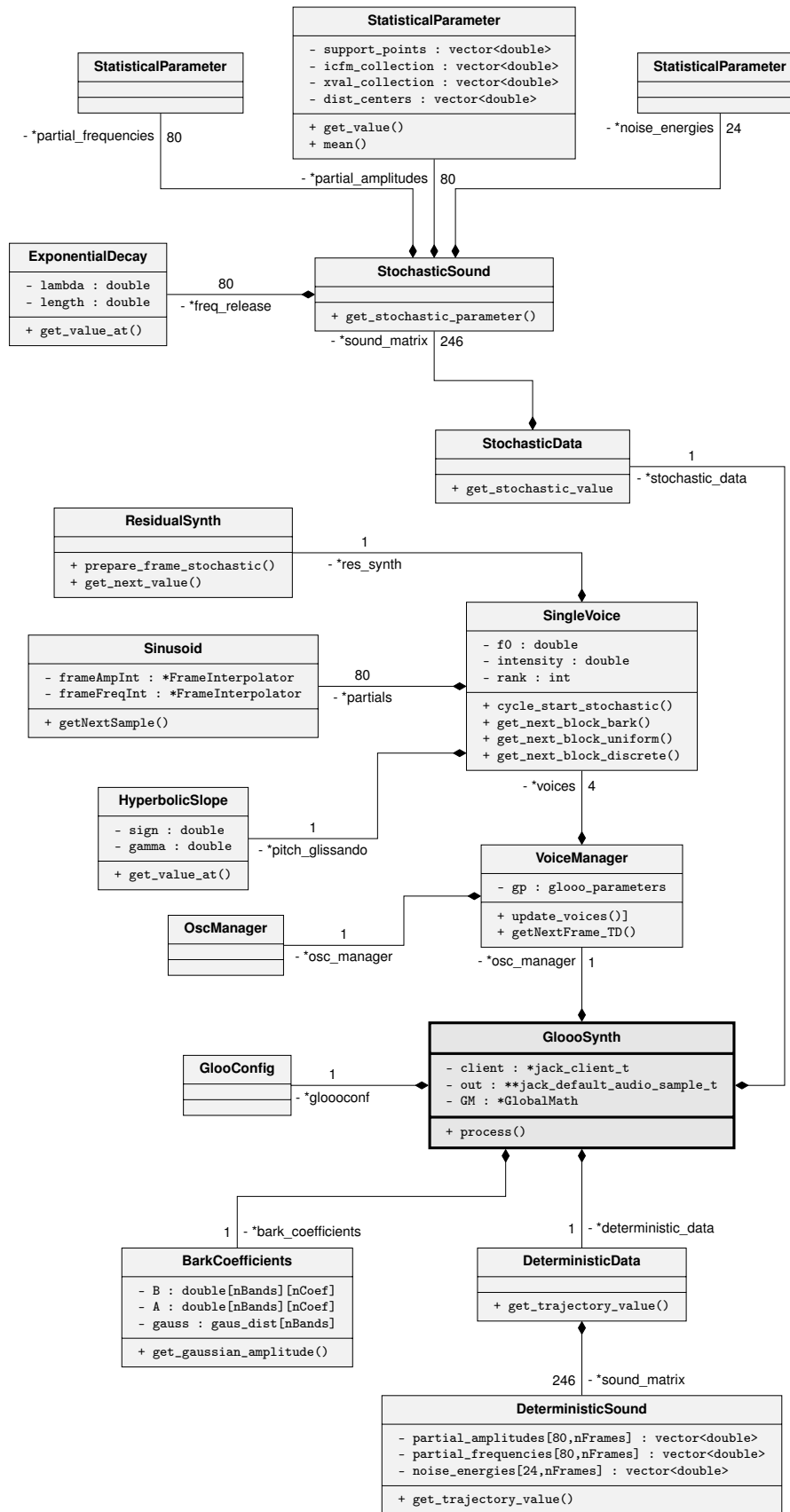


Figure 8.8: GLOOO synthesizer class diagram.

8.3.3 Voice Management

At this stage, the GLOOO synthesis engine is configured as a monophonic melody instrument with polyphonic capabilities. It therefore operates in a mode entitled *pseudo polyphony*. This mode implies two conditions:

1. Only one note can be controlled at a time.
2. Multiple notes may sound at a time.

These conditions are of importance for different articulation styles, especially for legato. In tightly coupled notes, the first note is still decaying, while the following note is entering the attack phase. For long release times and short inter onset intervals, it is very likely that more than two notes are sounding at the same time. Hence, there is always one active voice under direct control, whereas all other voices are in the release mode.

Management of the voices takes place in the `VoiceManager` class. The voice manager holds a vector of `SingleVoice` objects. The number of voices which can sound simultaneously is configured on startup, depending on the capabilities of the runtime system and the length of the releases in the instrument model. Four voices were used as a standard during development, testing and in the user study. If an onset is received when all available voices are already active, a typical voice stealing mechanism deactivates the oldest voice and uses it for the new sound. For this purpose, the single voices have the property `rank`, which is initialized as 0 for a newly activated voice and increased by one whenever a note onset is received. The voice manager class is aware of the voices' ranks and can thus handle the voice stealing.

The `SingleVoice` class acts as a state machine, with the states defined in the GLOOO data type `note_state`. It can be either `inactive` or in one of the active states `attack`, `sustain`, `release` or `transition`. All possible state transitions, including the voice stealing, are visualized in Figure 8.9.

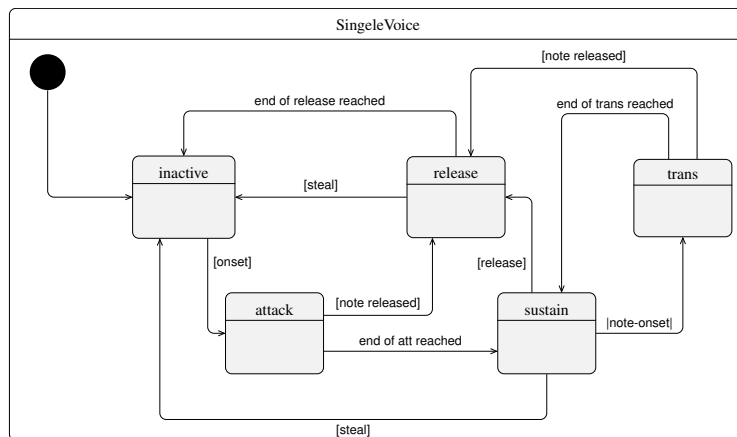


Figure 8.9: State diagram for the `single_voice` class.

8.3.4 Synthesis Process

The synthesis system allows the selection of different modes for the statistical synthesis, namely the mean mode and the Markovian inverse transform sampling mode (*MITSSy*), both introduced in Section 8.1. In addition, three dispersion modes, all explained in Section 8.2, can be chosen. Both aspects can be changed during runtime. Depending on their combination, the synthesis process is slightly different. In this section it is outlined for the *MITSSy* algorithm, using the *filter bank dispersion* mode, since this is the most complex variant.

The sequence diagram in Figure 8.11 illustrates the implementation of the algorithm to a reasonable depth and shows the involvement of the classes in the synthesis process. Every time the JACK processing function is called through an interrupt, the `synth` object calls the method `update_voices()` of the `VoiceManager` to process all note onset and offset information and enable or disable voices.

In the method `getNextFrame_TD()`, the `VoiceManager` then loops over all active voices, first setting their general synthesis parameters like pitch and intensity for the current synthesis frame, according to the control data. For each voice, the method `cycle_start_stochastic()` is called, in which the statistical values for the partial parameters of the recent frame are generated and their interpolation trajectories are pre-calculated.

Within the method `get_next_block_bark()`, the individual sinusoidal and residual output samples are obtained. Each voice loops over every partial, initially getting the contribution to each frequency band in the method `get_gaussian_amplitude()` of the `BarkCoefficients` class in a loop over all bands. Afterwards, each of the `nBuff` output samples of the partial is calculated in a loop and the weighting to each frequency band is applied.

The residual component is synthesized in a separate loop over all frequency bands, using the classes `ResidualSynth` and `BarkCoefficients` for generating the noise signal sample-wise.

SMS Rate

In order to use the deterministic trajectories, the exponential release models and the glissando transitions, the *SMS Rate* determines the ratio between sampling rate of the trajectories and the synthesis frame size N_{BUFF} . It depends on the sampling frequency $f_{s_{amp}}$ of the original audio files, the hop size of the spectral analysis and the sampling rate $f_{s_{synth}}$ of the synthesis system:

$$r_{SMS} = \frac{\frac{N_{BUFF}}{f_{s_{synth}}}}{\frac{N_{HOP}}{f_{s_{amp}}}} \quad (8.10)$$

Number of Partial

Figure 8.10 shows the maximum number of partials to be synthesized in dependency of the system's sample rate.

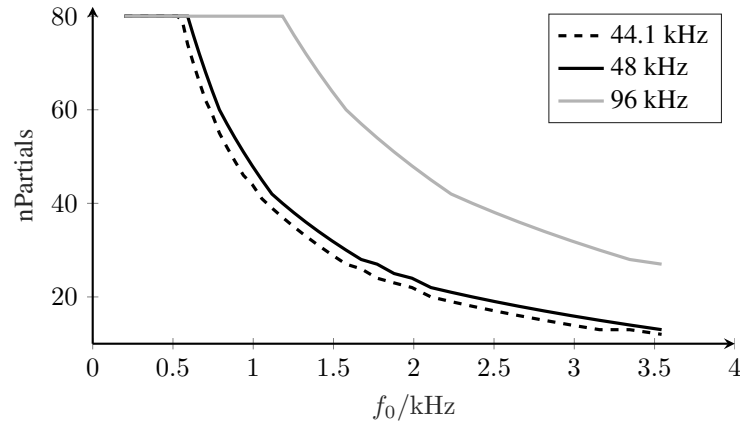


Figure 8.10: Number of synthesized partials for different fundamental frequencies and sampling rates.

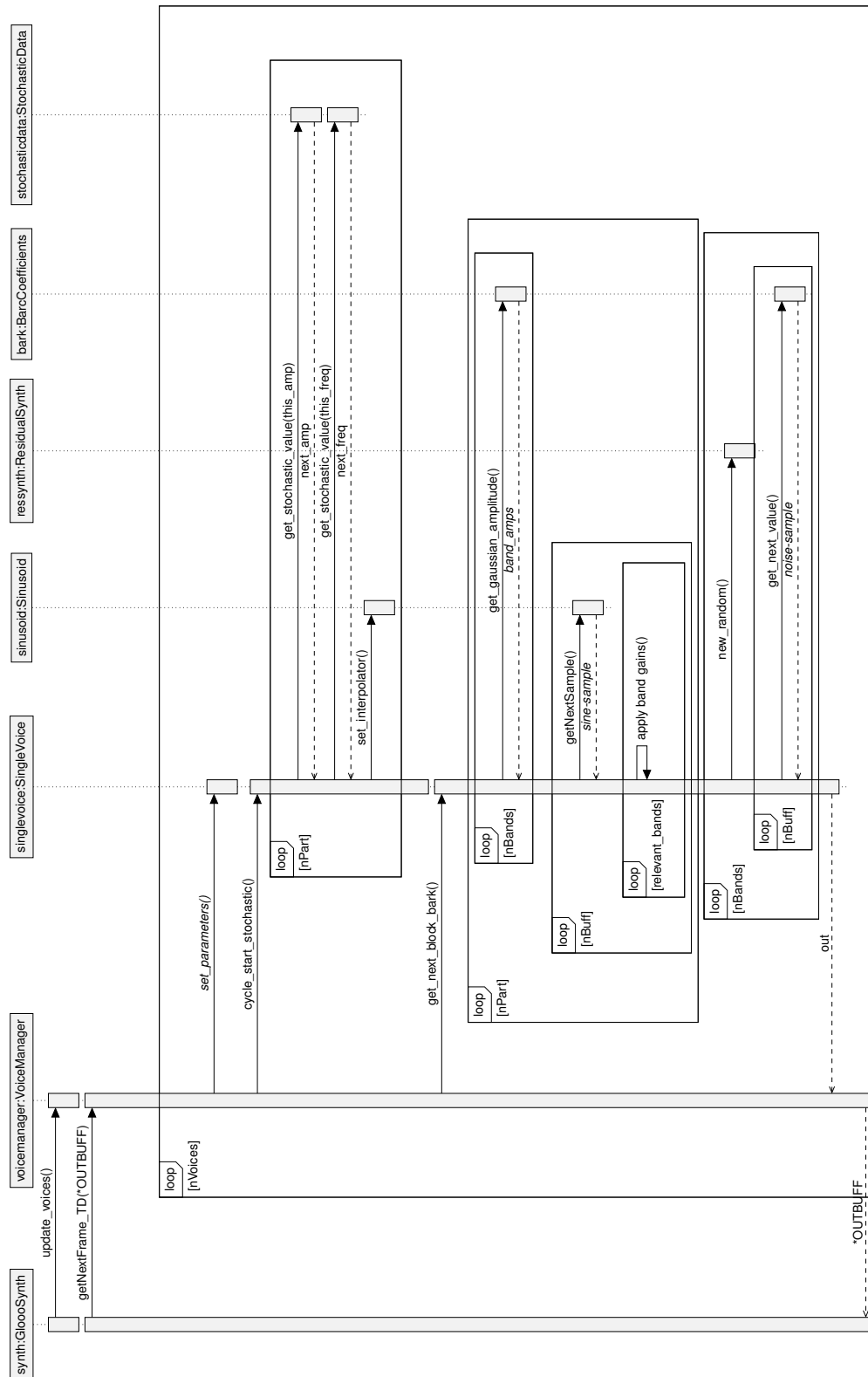


Figure 8.11: Sequence diagram for the JACK callback process in the filter bank dispersion mode.

8.3.5 GLOOO Configuration

The class `GloooConfig` reads all configuration parameters from a YAML file, which is passed to the program as a command line argument. An example file for regular use can be found in Appendix III. It contains the basic properties of the synthesis system which are set once during startup and can not be changed during runtime.

Paths and Files

The model to be used is passed to the synthesis software as a command line argument. This section allows the use of different assignment and filter files within this model.

Table 8.3: Assignment file and filter model in the configuration file.

Parameter	Description
<code>assignment_file</code>	The assignment file defines which subset of a full model is read during startup. Items in the sample library are assigned to combinations of pitch and intensity. When testing specific aspects, minimized versions like the 8-sample <code>sample-assignment_SMALL</code> can be used for faster start.
<code>bark_file</code>	This file holds the coefficients of the Bark scale filters and the parameters of their Gaussian counterparts. Currently, individual files are required for different sampling rates.

Basic Synthesis Settings

The general synth parameters relate to the performance of the runtime system. The standard values can be decreased to allow a smooth operation on lightweight hardware.

Table 8.4: Basic synthesis settings the configuration file.

Parameter	Description
<code>num_voices</code>	The maximum number of voices for the pseudo polyphony.
<code>num_partials</code>	The maximum number of partials to be used in synthesis.
<code>synth_mode</code>	The synthesis mode can be changed during runtime. This parameter defines which data sets should be read during startup. The full system requires the mode <code>dual</code> , with both the deterministic and the stochastic data set. For limited use with faster startup, single parts of the data can be loaded.

Communication Settings

In the OSC section, users can define ports for incoming and outgoing OSC messages. OSC and MIDI activity can be disabled, globally. The recent version of the software does not implement MIDI, since the protocol does not fulfill the requirements on the control signals. The GLOOO synthesis system still sticks to the MIDI scale for pitch, yet uses a decimal format. Future versions will most likely rely on MIDI to OSC bridges, if this protocol is needed.

Table 8.5: Communication parameters in the configuration file.

Parameter	Description
<code>osc_port_in</code>	The OSC port to receive control data.
<code>osc_port_out</code>	An outgoing OSC port for sending data to peripheral software. This option is not in use at this point.
<code>receive_osc</code>	When set to 0, the synthesizer will ignore incoming OSC messages.
<code>receive_midi</code>	When set to 0, the synthesizer will ignore incoming MIDI messages.

Debug Outputs

This section of the configuration file tells the software which information to output for debugging and evaluation. Some of these settings radically impair the performance of the system through writing operations at a high rate.

Table 8.6: Debug output parameters in the configuration file.

Parameter	Description
<code>partial_trajectories</code>	If set to 1, the synth will write the amplitudes of all partials to a text file.
<code>partial_output_path</code>	Defines the output path for the text file with the partial amplitudes.
<code>control_input</code>	If set to 1, the software displays incoming control signals in terminal.
<code>statistical_data</code>	If set to 1, information on the stochastic synthesis will be displayed in the terminal.

8.3.6 Synthesis Parameters

The synthesis software receives control input through OSC messages, processed in the class `OscManager`. Figure 8.12 shows all parameters of the synthesis system which can be controlled during runtime in this way. Tables 8.7 to 8.10 list the OSC paths and arguments used in the implementation. A more detailed list of all controls is included in the online user guide.

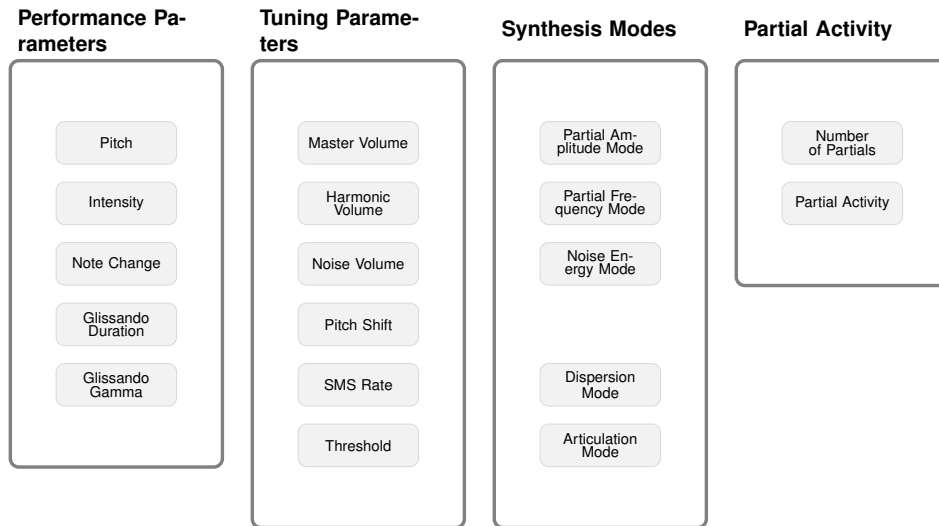


Figure 8.12: Input parameters of the GLOOO synthesis program.

The Performance Parameters are used for the actual expressive control, including the continuous parameters *Pitch* and *Intensity*. The additional parameter *Note Change* is used for legato transitions. Duration and slope of the glissando can also be changed as performance parameters.

Tuning Parameters include the master volume and the harmonic-to-noise ratio, as well as the global pitch shift, the SMS Rate and the trigger threshold for note onsets. New voices are only launched if the intensity exceeds the threshold, which helps integrating different control devices.

The individual Synthesis Modes of spectral modeling parameters can be changed during runtime. Partial amplitudes can be used in the *stochastic* mode or in the *mean* mode. Partial frequencies offer the modes *mean* and a *fixed* harmonic mode. Noise band energies can be synthesized in the *mean* mode or a *stochastic* mode. The dispersion modes *uniform*, *Bark* and *discrete* can be selected, as well as the articulation modes *legato* and *glissando*.

Two parameters offer control over the partial activity during synthesis. *Number of Partial* sets the maximum number partials to be synthesized as an integer value ranging from 0 to the maximum possible number, in this case 80. In addition, *Partial Activity* offers control over the activity of individual partials as a switch.

Table 8.7: Performance Parameters.

Parameter	OSC path	Argument 1	Argument 2
Intensity	/intensity	f: linear (0-1)	
Pitch	/pitch	f: decimal MIDI	
Legato Note change	/note	i: MIDI pitch	i: MIDI velocity
Glissando Duration	/gliss_duration	f: seconds (0...5)	
Glissando Gamma	/gliss_gamma	f: (0...1.5)	

Table 8.8: Tuning Parameters.

Parameter	OSC path	Argument 1	Argument 2
Global Volume	/master_volume	f: linear (0...2)	
Harmonic Volume	/tonal_volume	f: linear (0...1)	
Noise Volume	/noise_volume	f: linear (0...1)	
Pitch Shift	/pitch_shift	f: multi (0.1...10)	
SMS Rate	/sms_rate	f: multi (0.1...2)	
Trigger Threshold	/threshold	f: (0...1)	

Table 8.9: Synthesis Modes.

Parameter	OSC path	Argument 1	Argument 2
Partial Amplitude Mode	/part_amp_mode	s: (s,m)	
Partial Frequency Mode	/part_freq_mode	s: (m,f)	
Noise Energy Mode	/noise_energy_mode	s: (s,m)	
Dispersion Mode	/disp_mode	s: (u,b,d)	
Articulation Mode	/art_mode	s: (l,g)	

Table 8.10: Partial Activity.

Parameter	OSC path	Argument 1	Argument 2
Number of Partials	/n_partials	i: max index (0...80)	
Partial Acticity	/partial_active	i: partial index	i: on (0,1)

8.4 Measurements

The measurements presented in this section investigate basic qualities of the statistical spectral modeling approach on a signal level. If the algorithm is capable of expressive sound synthesis, as defined in Section 3.4, it should react to specific input sequences with a particular behaviour. For this purpose, the real-time synthesis software is controlled with pre-defined sweeps of the parameters pitch and intensity. These sweeps are sent as OSC streams from a Python-based trajectory generator, which is part of the synthesis software repository (von Coler, 2020c). The responses of the synthesis system to these clearly defined input sequences are recorded. Evaluations focus on the deterministic component. Due to the nature of the synthesis software, all partials can be recorded separately, using the command line tool `jack_capture` with 80 input channels. Results of the measurements are evaluated through comparison with expected behavior to deliver a proof of concept for the implementation.

8.4.1 Pitch Sweeps

As introduced in Section 2.4.1, the amplitude of an individual partial depends not only on the global intensity, but also on its instantaneous frequency. More precisely, it depends on the fundamental frequency of the sound played, the partial index and the frequency response of the instrument's resonating body. When the pitch is modulated, this results in a modulation of the partial amplitudes and thus of timbre. This phenomenon, referred to as spectral envelope modulations, plays an important role in the timbral characteristics of a string instrument like the violin.

In order to prove the presence of spectral envelope modulations, the synthesis system is driven with pitch sweeps of one octave, at four different stationary intensity levels. Pitch is moving from MIDI pitch 55 to MIDI pitch 67. Intensity levels are set to MIDI velocity 20, 50, 80 and 120. Figures 8.13 to 8.15 show the amplitudes of the first three partials as a function of their frequency for the highest intensity. All three plots show distinct peaks and dips, related to the formants of the violin's resonant body and other components.

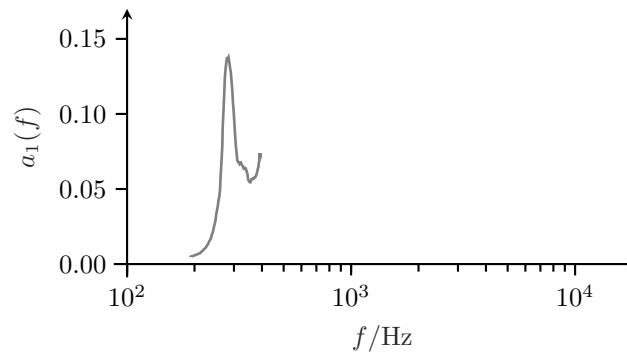


Figure 8.13: Amplitude over frequency for the first partial during a pitch sweep of one octave.

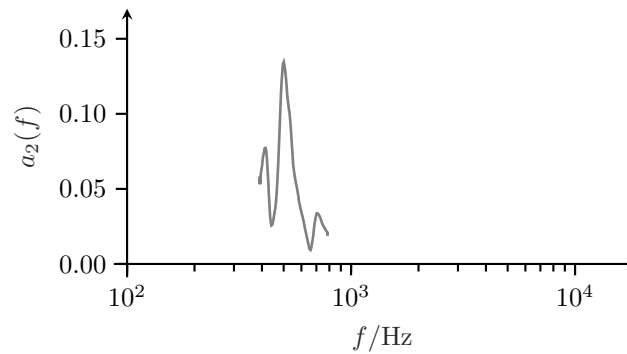


Figure 8.14: Amplitude over frequency for the second partial during a pitch sweep of one octave.

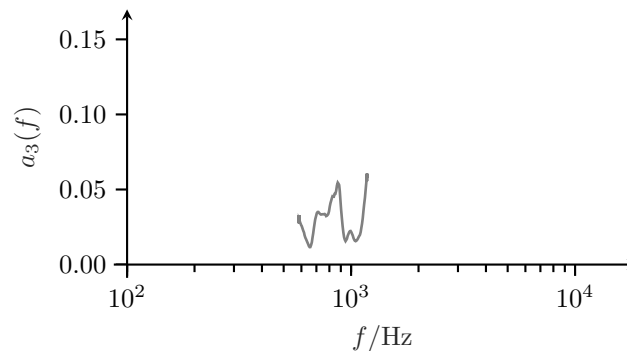


Figure 8.15: Amplitude over frequency for the third partial during a pitch sweep of one octave.

Amplitude Response Approximation

The sweep responses of all 80 partials can be combined, resulting in an approximation of the amplitude response of the violin. Figure 8.16 shows the combined responses for the sweep at the highest intensity with a logarithmic frequency scale. The higher the frequency, the more the responses of the neighboring partials overlap, resulting in ambiguities through variations for each partial and a thicker combined curve. Nevertheless, the approximation shows distinct formants and has the properties of an actual amplitude response.

For validating the approximated amplitude response in Figure 8.16, it is compared to measurements of the acoustical properties of an actual instrument. Alonso Moral and Jansson (1982) measured the input admittance of 24 violins from a Scandinavian violinmakers' competition. A model by Andrea Guarneri, dated approximately 1690, was included as reference. The input admittance was measured in two positions on top of the violin bridge and perpendicular to the top plates of the violins. The input admittance plot for the Guarneri violin is shown in Figure 8.17. Peak frequencies of four resonances were accurately measured, including the Helmholtz resonance, labeled A0, the main vibrations in the top plate, marked T1, as well as in-phase vibrations of the two plates with the labels C3 and C4. A rise around 3 kHz is associated with the bridge resonance. The prominent peak labeled F was not mapped by Alonso Moral and Jansson. For all violin models, the mean frequency was measured at 272 Hz for A0 (std = 11 Hz), at 403 Hz for T1 (std = 22 Hz), at 500 Hz for C3 (std = 27 Hz) and at 686 Hz for C4 (std = 54 Hz).

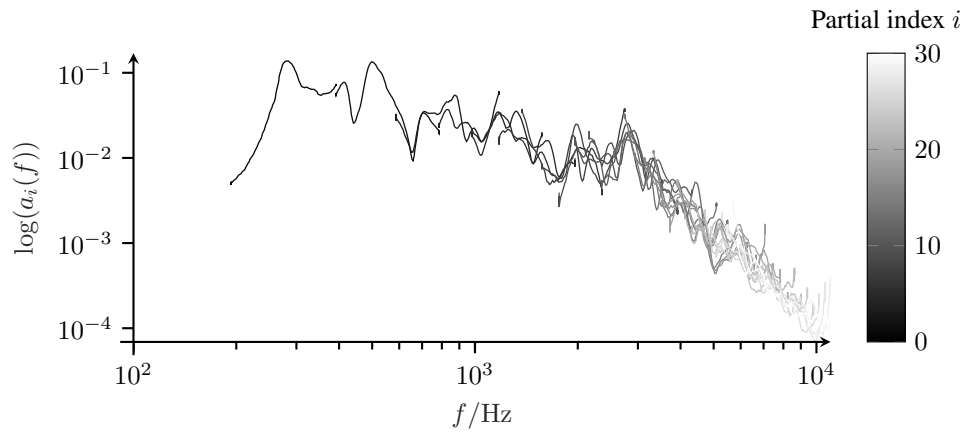


Figure 8.16: Approximation of the frequency response through a pitch sweep.

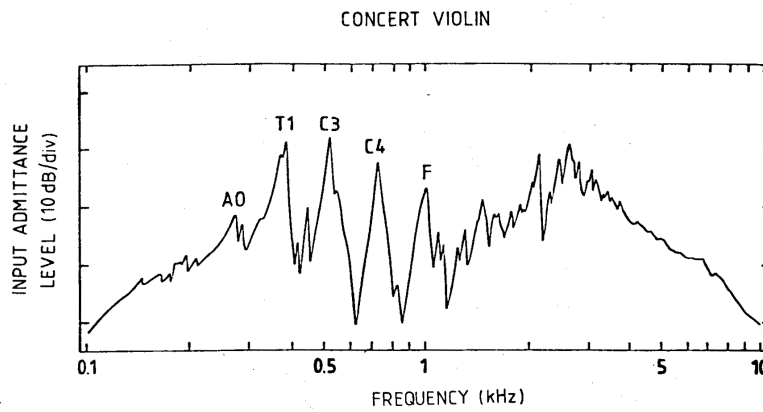


Figure 8.17: Input admittance at the bass bar side of the Andrea Guarneri violin (Alonso Moral & Jansson, 1982).

The pitch sweep approximation in Figure 8.16 shows the characteristics of the admittance plot, with a similar formant structure but less pronounced peaks and dips. For a more thorough investigation, Table 8.11 lists the most prominent modes of violins, based on a summary by Curtin and Rossing (2010). This work also includes the measurements by Alonso Moral and Jansson (1982). Naming of violin modes is not consistent throughout the literature and the labels used in this publication differ from those used by Alonso Moral and Jansson. The Helmholtz resonance, also known as f-hole resonance, labeled A_0 is typically located around 285 Hz. Modes between 400 Hz and 700 Hz, labeled C_n , refer to the corpus, respectively to motion of the plates, and are considered important for the individual character of the instrument. A_3 is associated with the lateral air motion. The violin formant F , or bridge hill, is located between 2000 and 3000 Hz.

Table 8.11: Main resonances of a violin body, revisited by Curtin and Rossing (2010).

Label	Frequency	Description
A_0 :	285 Hz	<i>f-hole resonance (Helmholtz resonance)</i>
C_1	405 Hz	<i>symmetrical motion of both plates</i>
C_2 (T1):	460 Hz	<i>strong motion of the top plate (main wood)</i>
C_3 :	530 Hz	<i>strong two-dimensional motion of both plates</i>
C_4 :	700 Hz	<i>symmetrical motion of both plates</i>
A_3 :	1000 - 1100 Hz	<i>lateral air motion</i>
F :	2000 - 3000 Hz	<i>violin formant, bridge hill</i>

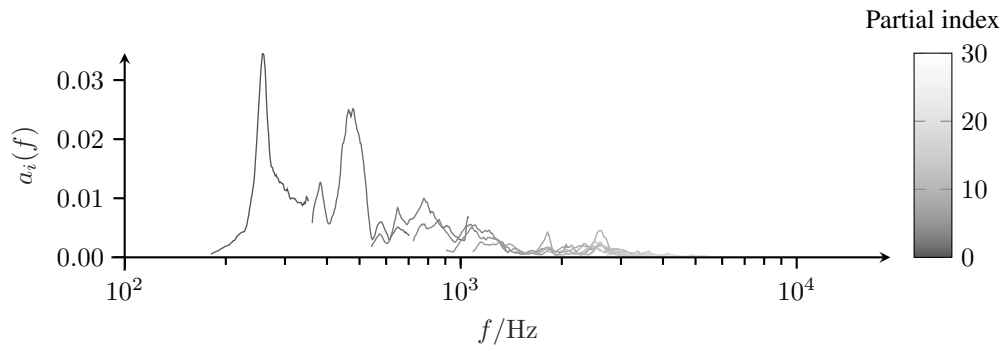
For an expressive spectral synthesis system, the above presented characteristics should be represented in the amplitude response approximated through the pitch sweeps. All main resonances listed in Table 8.11 and visualized in the input admittance plot in Figure 8.17 can be found in the results of the pitch sweep measurements in Figure 8.16.

For a better investigation, Figure 8.18 shows the approximated amplitude responses for all four intensity levels in individual plots. Differently scaled linear ordinates are chosen, allowing a clear identification of peaks and a better comparison of the four sweeps. Higher intensities show a relative increase of high frequency content. The bridge hill is more pronounced for higher intensities, which results in a brighter timbre. This relation, however, can be better investigated through intensity sweeps, presented in the following section.

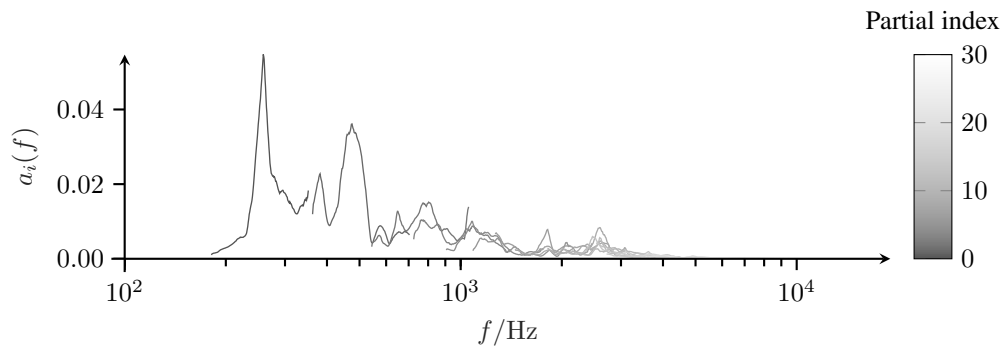
The same pattern of resonances can be found in the approximated amplitude responses for all intensities. However, they are most recognizable in Figure 8.18(d). Table 8.12 lists the frequencies of the most prominent peaks for the sweep at highest intensity and associates them with the resonances specified by Alonso Moral and Jansson (1982) and Curtin and Rossing (2010). Evidently, all major resonances are represented by the synthesis system. A_0 , C_1 and C_2 show only slight deviations from the references. The peaks between 660 Hz and 800 Hz are more ambiguous but associated with the modes C_3 and C_4 . A clear peak is found within the range of the lateral air motion A_3 . The bridge hill, or violin formant, shows four distinct peaks between 1800 Hz and 3000 Hz.

Table 8.12: Main resonances from the pitch sweep measurements in Figure 8.16.

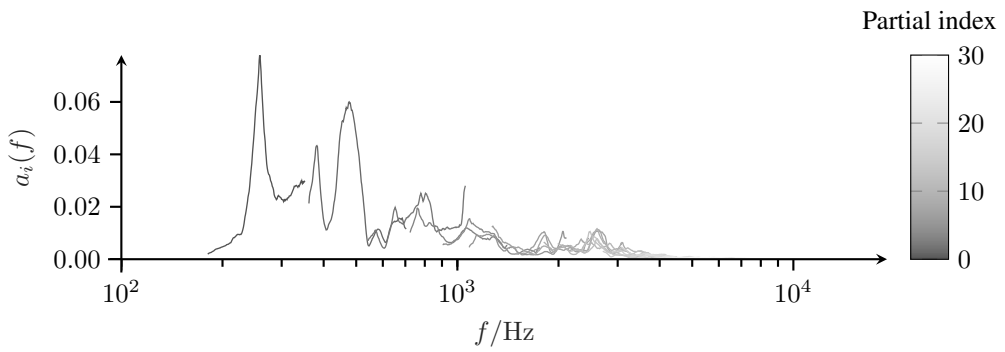
Frequency	Association
259 Hz	<i>Helmholtz resonance (A_0)</i>
380 Hz	<i>symmetrical motion of both plates (C_1)</i>
457 Hz	<i>motion of the top plate (C_2)</i>
660 Hz, 760 Hz, 800 Hz	C_3, C_4
1098 Hz	<i>lateral air motion (A_3)</i>
1830 Hz	<i>bridge hill (F 1)</i>
2070 Hz	<i>bridge hill (F 2)</i>
2627 Hz	<i>bridge hill (F 3)</i>
3000 Hz	<i>bridge hill (F 4)</i>



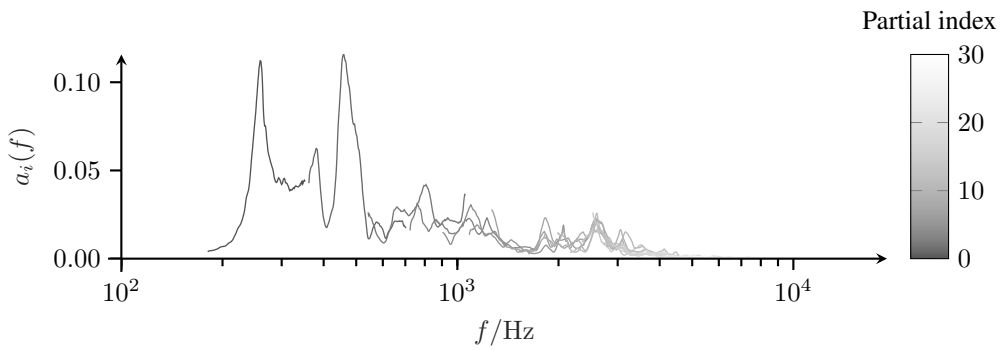
(a) Low intensity (pp).



(b) Medium-low intensity (mp).



(c) Medium-high intensity (mf).



(d) High intensity (ff).

Figure 8.18: Approximations of the frequency response through spectral envelope modulations of 30 partials by one octave sweep.

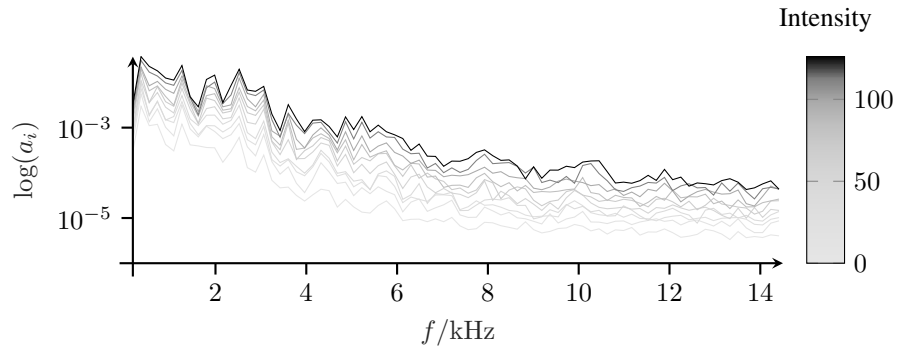
8.4.2 Intensity Sweeps

Measurement Method

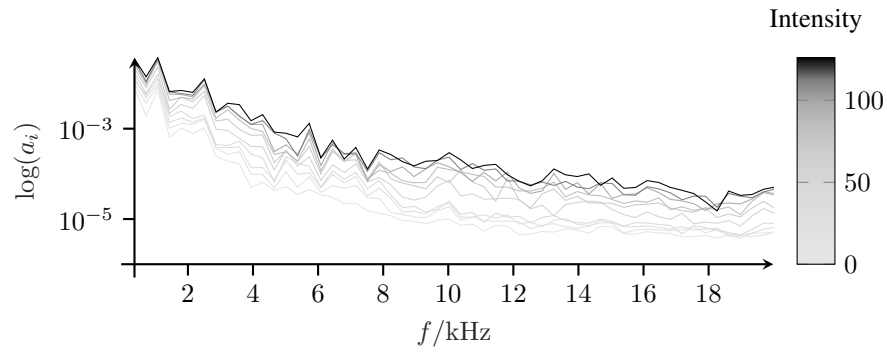
The dependence of an instrument's timbre on the intensity is an important aspect, not only in expressive sound synthesis. For most acoustical instruments, the brightness, respectively the relative amount of high frequency energy, increases with the intensity of excitation. This relationship could already be observed in the pitch sweeps of the previous section. In order to investigate this aspect more precisely, the response of the synthesizes system to sweeps in intensity is measured at different pitches. As for the pitch sweeps, each partial is recorded separately. Hence, the spectral envelope of the harmonic component can be expressed through the partial amplitudes at multiples of the fundamental frequency.

The system's response to intensity sweeps is captured at four pitches, namely 180 Hz, 358 Hz, 715 Hz and 1429 Hz. Each sweep is then sampled at ten instances, equally spaced between the minimum and maximum intensity. Figure 8.19 shows the resulting harmonic spectral envelopes during intensity sweeps at these instances for all four pitches. MIDI values are used for the intensity, ranging from 1 to 127. The maximum frequency for the representation is limited to 20 kHz. For each pitch, the envelopes have a different range and a different number of support points, hence a different resolution. For 180 Hz, all 80 partials can be synthesized, ranging up to 14.4 kHz. The maximum number of partials for 358 Hz is 55. A total of 27 partials is captured at 715 Hz and only 13 partials at 1429 Hz.

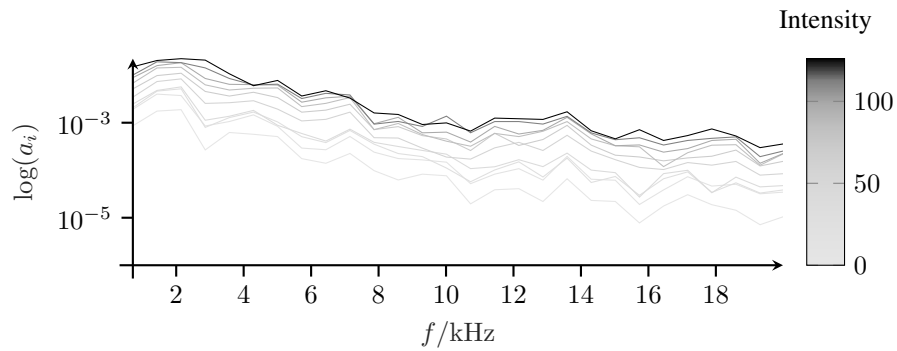
The harmonic spectral envelopes do not represent the resonances of the violin as clearly as the pitch sweeps. Especially for high pitches, with only few support points within the Nyquist frequency, the resolution of the envelopes is impaired. However, they are well suited for evaluating the overall distribution of spectral energy. For all four pitches, the relative increase of high frequency content is recognizable for higher intensities.



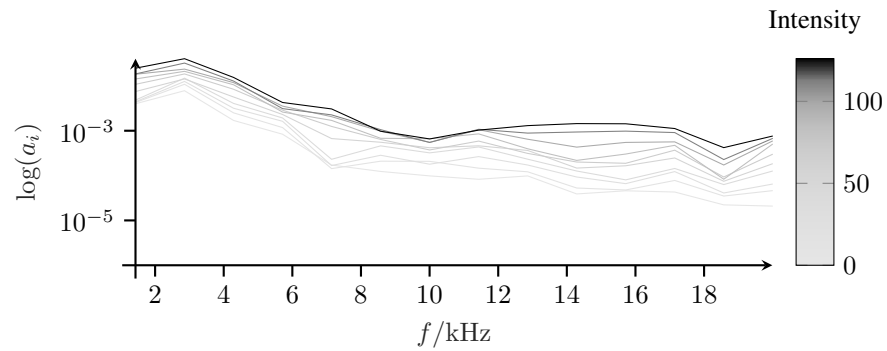
(a) 180 Hz.



(b) 358 Hz.



(c) 715 Hz.



(d) 1429 Hz.

Figure 8.19: Harmonic spectral envelopes for four different pitches, sampled at 10 instances during intensity sweeps.

Harmonic Spectral Centroid

The harmonic spectral centroid is used to quantify the relation between intensity and timbre. The spectral centroid is a common audio feature for describing the distribution of energy within the spectrum, usually defined as the center of mass (Lerch, 2012). With slight variations in implementation, it is in general calculated as the first moment of the spectral distribution. Based on the spectral centroid, the harmonic spectral centroid (HSC) regards only the amplitudes a_i of the N partials. It can be calculated with the partial indices i , resulting in a pitch independent measure, or with the partial frequencies f_i as weighting factors for the spectral distribution of the partials. The latter version is chosen for the evaluation of the measurements in this section:

$$\text{HSC} = \frac{\sum_{i=1}^{i=N} f_i a_i}{\sum_{i=1}^{i=N} a_i} \quad (8.11)$$

The HSC is calculated for the above introduced intensity sweeps at 50 instances, equally sampled across the full intensity range. Figure 8.20 shows the HSC as function of the intensity for four different pitches. For 180 Hz and 358 Hz, the HCS is lower in average than for 715 Hz and 1429 Hz. Besides that, all trajectories show a similar behavior, a quasi monotonic increase in the HSC with increasing intensity. The difference in HSC between the lowest and highest intensity yields 419 Hz for a pitch of 180 Hz, 631 Hz for a pitch of 358 Hz, 781 Hz for a pitch of 715 Hz and 827 Hz for a pitch of 1429 Hz.

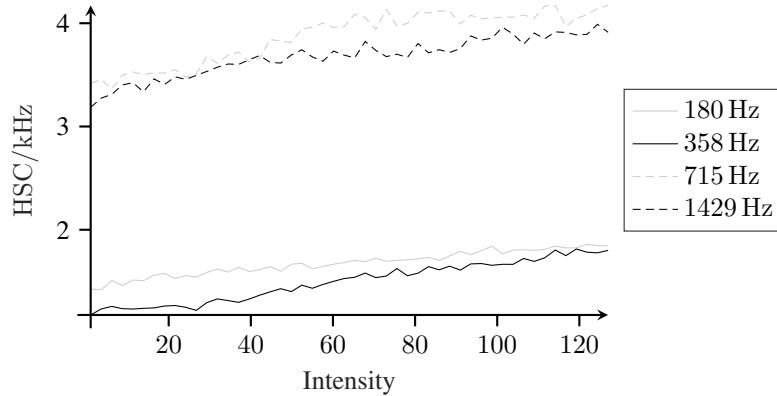


Figure 8.20: Harmonic Spectral Centroid as function of intensity at four different pitches.

A simple linear regression between intensity int and HCS for all pitches shows a significant relationship for all cases ($p < 0.001$) with $F(1, 48) = 931$ for 180 Hz, $F(1, 48) = 1730$ for 358 Hz, $F(1, 48) = 243$ for 715 Hz and $F(1, 48) = 411$ for 1429 Hz. The predicted HCS is equal to $1477.5 + 3.095 \text{ int}$ for 180 Hz, $1170.9 + 5.178 \text{ int}$ for 358 Hz, $3505.6 + 6.282 \text{ int}$ for 715 Hz and $3406 + 5.711 \text{ int}$ for 1429 Hz. As expected from the system, an increase in intensity results in a significant increase in the Harmonic Spectral Centroid which causes a change in timbre, more precisely in an increase in brightness.

Control & Mapping

This chapter is dedicated to the aspects of control and mapping in the GLOOO project. This includes the design and development of the BINBONG, a haptic musical interface for the synthesis system. The original idea, first design studies and the development of the first version are outlined in Section 9.1, followed by a more detailed view at the second version in 9.2. The receiver and mapping software, realized in Pure Data, as well as the mapping between the interface, synthesis and spatialization are explained in Section 9.3.

9.1 BINBONG MK I

9.1.1 Motivation and First Experiments

There are numerous ways to control electronic and digital musical instruments. Depending on the context, this includes standard input devices like the computer keyboard and mouse, sequencers and algorithms, data streams for sonification, conventional MIDI controllers and individual devices for direct control. All approaches have their field of application, rooted in musical genres and practices. However, the key to a truly expressive electronic musical instrument is an expressive interface. It is thus only consequent to pair sound synthesis algorithms with such interfaces to unleash their full potential and turn them into a musical instrument.

The development of the BINBONG, the haptical musical interface presented in this chapter, started in 2014 and was motivated by the work on the expressive synthesis engine, which was in an early stage. Initially, the device was a side project, intended for general application, but over time the features of the controller were developed with regard to the synthesis software. First experiments were inspired by a small setup which Edgar Berdahl left in Berlin after his time as a visiting researcher at the Audio Communication Group. It was a single board computer with one force-sensing resistor (FSR) attached to it, programmed to control the playback speed of a

previously recorded sample.

From the beginning, the design of the BINBONG was based on the hypothesis that a general muscular tension, and thus force, is one way to enable expressive control. As introduced in Section 2.4, the use of vibrato on acoustical instruments is based on such muscular tension. Another important concept in the presented device is the decoupling of excitation and frequency control, which is known to be beneficial for expressive musical interfaces, as explained in Section 4.2. The very first design studies do not feature a concept for excitation at all, but focus on an expressive mechanism for pitch control.

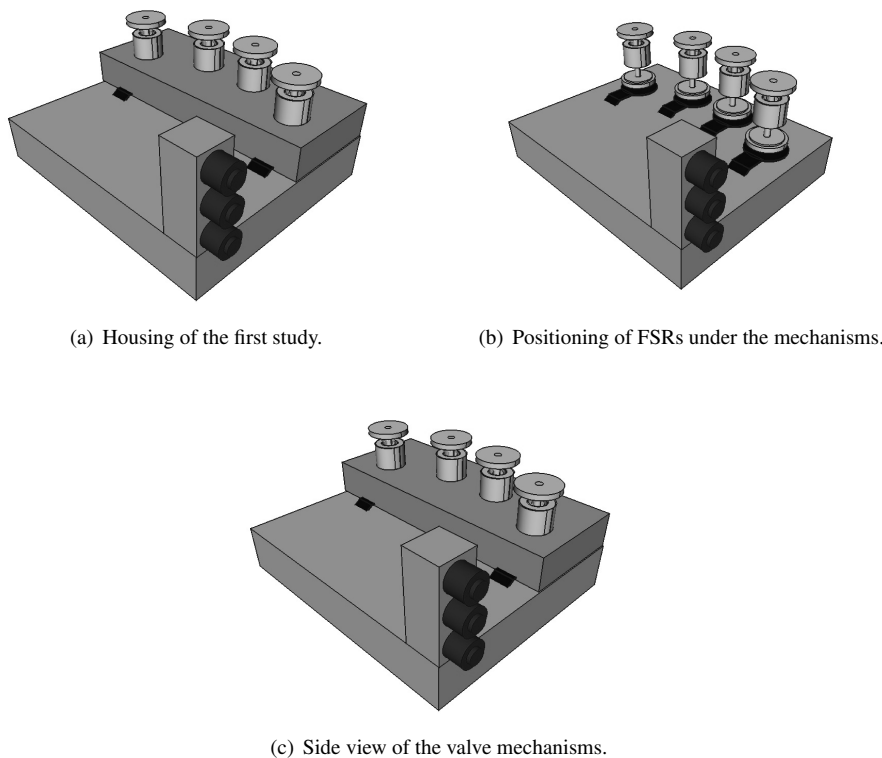


Figure 9.1: First design study for a tabletop device with four FSRs and three octave switches.

After experiments with force-sensing resistors on a breadboard at SIM Berlin, first design studies for the controller were drawn. The very first sketch was a tabletop device, shown in Figure 9.2. It already features the four mechanisms, entitled *valve mechanisms*, for the binary pitch selection with underlying force sensing resistors (FSR) for enhanced expressivity. Three octave switches extend the pitch range of the concept. A similar use of FSRs is very common in the design of comparable interfaces for musical expressivity (M. T. Marshall et al., 2009). Although the tabletop design has certain advantages when used in a setup with additional electronic instruments, the stationary concept would limit the expressive capabilities.

A first hand-held design, also without means for controlling the excitation, is shown in Figure 9.1. It features the same mechanisms and sensors as the table-top version and can be considered a predecessor of the following design studies.

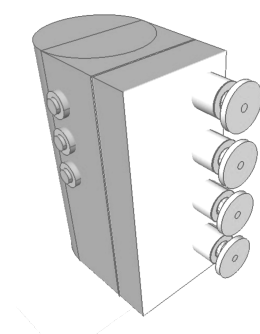
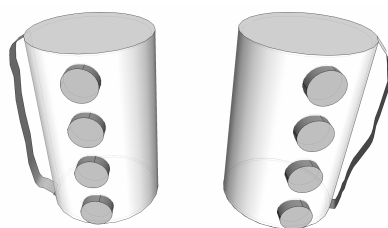


Figure 9.2: First hand-held design without excitation.

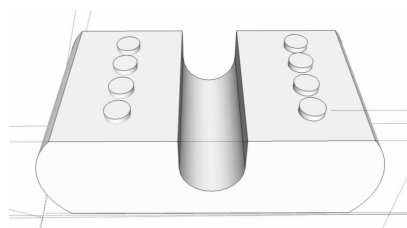
9.1.2 Design Studies

The first actual version of the controller was developed within a master's thesis by Gabriel Treindl (Treindl, 2016) in a cooperation between Technical University Berlin and the *Staatliches Institut für Musikforschung, SIM* (Federal Institute for Musicology). Major goal of this stage was to find a configuration that would feature at least four valve mechanisms with additional excitation control and to evaluate this concept in a user study. Tom Lerch, restorer for wind and percussion instruments and head of the restoration workshops at the SIM, contributed major aspects to the design of the housing and the implementation of mechanical components. The project would not have been possible without his skills and ideas, especially the transfer of knowledge from traditional instrument making.

Gabriel Treindl explored different ways of combining four or eight valve mechanisms, some under consideration of principles for excitation, in hand-held devices. Figure 9.3 and Figure 9.4, show different concepts. Four designs feature eight valves, including an uncoupled two-handed version and two-handed designs without additional excitation, as well as a wind controller. Most important is the first BINBONG design in Figure 9.4(c), which was chosen for implementation. It already features an excitation pad, as it is used in the following prototype.



(a) Two-handed design (separated).



(b) Two-handed design with 8 fingers.

Figure 9.3: Design studies I by Gabriel Treindl (von Coler, Treindl, Egermann, & Weinzierl, 2017).

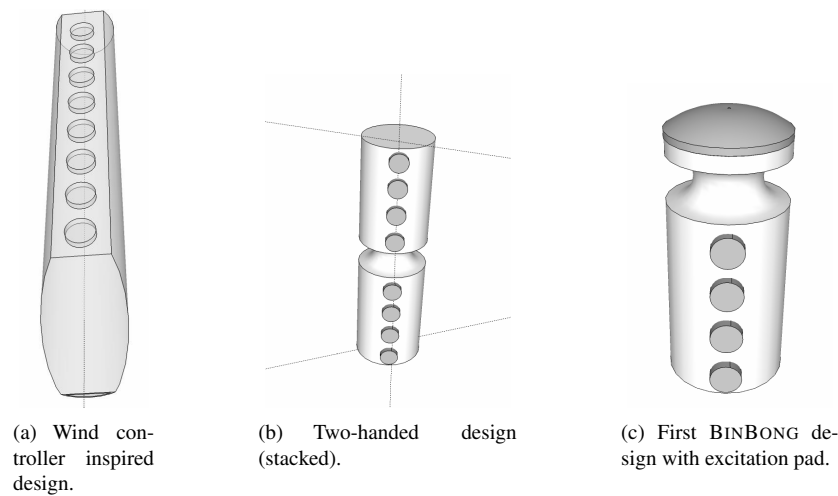


Figure 9.4: Design studies II by Gabriel Treindl (von Coler, Treindl, Egermann, & Weinzierl, 2017).

9.1.3 First Prototype

The valve mechanisms, made of brass and designed and manufactured by Tom Lerch, are the key elements for the further development of the BINBONG. Figure 9.5 shows a schematic representation. Noteworthy are the custom molded silicone pads between valves and the force sensing resistors. Using knowledge from hearing aid acoustics, Gabriel Treindl optimized their composition to provide a soft action point, allowing expressive control.

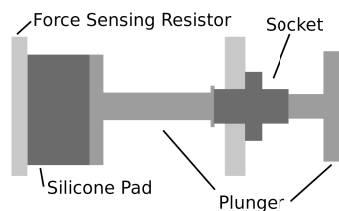
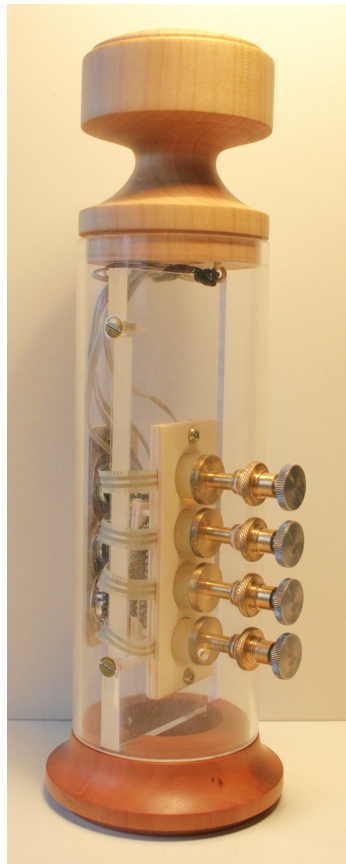


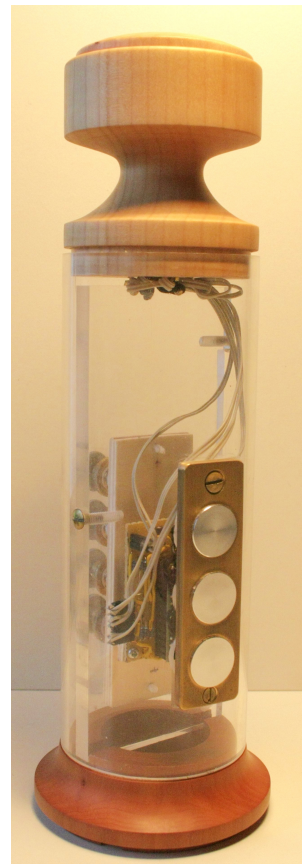
Figure 9.5: Valve mechanism.

The first prototype, shown in Figure 9.6, features four valve mechanisms for pitch selection, an excitation pad and three octave switches in a transparent acrylic cylinder. Excitation pad and foot are made of turned wood. Four FSRs are located under the wooden pad, sensing force and location of pressure. Valve mechanisms and socket for the octave switches are made of brass.

Processing of the raw sensor data is implemented on a *Teensy 3.1* development board, with minimalistic additional circuitry. It has all eight force-sensing resistors connected directly to the analog to digital converters of the board, four from the valve mechanisms, four from the excitation pad. Three piezo sensors for the octave switches are connected to the digital inputs of the Teensy. Sensor data is pre-processed on the board and sent to a computer using MIDI over USB. The device is thus corded. The master's thesis by Gabriel Treindl features in depth details on the production and programming of the prototype (Treindl, 2016).



(a) Front view, showing the valve mechanisms.



(b) Rear view, showing the octave switches.

Figure 9.6: Front (a) and rear (b) view of the BINBONG MKI.

User Study

In order to evaluate the first prototype, a user study was conducted. The study aimed at the general usability of the interface in simple tasks without musical context. Visual stimuli were presented to the participants, defining a sequence of combinations to be pressed. Error rate and response time were measured and compared to the performance on a MIDI keyboard. Overall, the BINBONG showed a higher error rate but no increase in the response time.

According to responses from a survey, participants appreciated the novel concept. Details on the study and the results are included in the related AES convention paper (von Coler et al., 2017).

9.2 BINBONG MK II

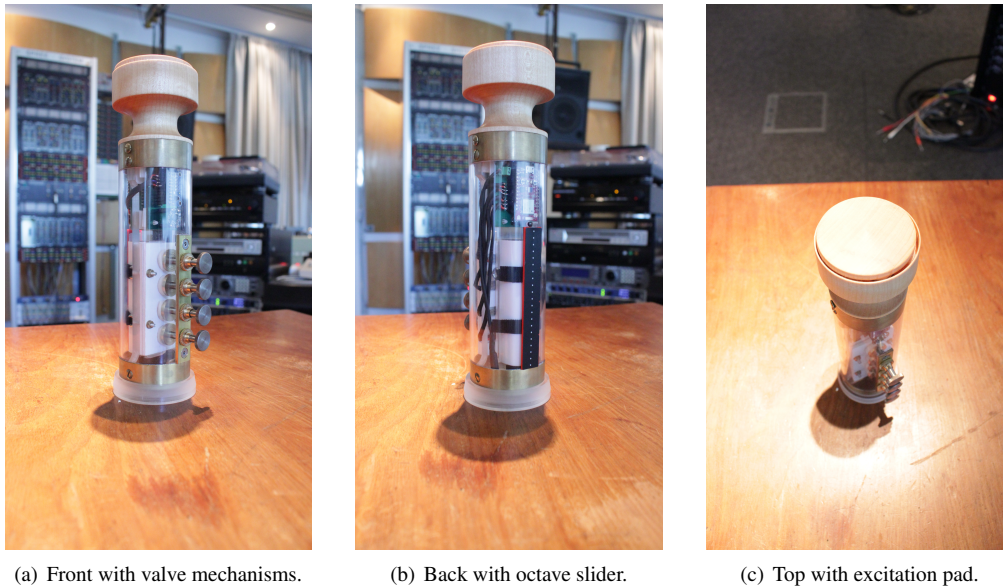


Figure 9.7: The BINBONG MKII.

9.2.1 Improvements to the First Version

The first version of the BINBONG served as a proof of concept and was appreciated by most users. However, the prototype left numerous possibilities for improvement, in order to make it a proper musical interface. Thus, a second version was developed, based on the experiences with the first prototype. Valve mechanisms and excitation pad should be included in future versions, since they showed a great potential for expressive control.

Besides changes in the dimensions of the housing, some materials and the arrangement of the control elements, two major aspects had to be added. The device should communicate with the receiving computer wirelessly, in order to increase the possibilities for expressive gestures. Further, the device's orientation in space should be tracked with an additional sensor unit. This already aimed at the application of the BINBONG as an interface for spatial sound synthesis.

The second version was implemented by Anton Schmied in his master's thesis (Schmied, 2018), continuing the cooperation with the SIM, respectively Tom Lerch. In contrast to the development process of the first prototype, which was rather explorative, this phase had clear goals, based on previous experiences and findings. An important aspect was the possibility to produce multiple units, making use of printed circuit boards and requiring a thorough documentation. Figure 9.7 shows one of five completed units of the second version, again with four valve mechanisms and the wooden pad, but with a different concept for octave selection and slightly altered dimensions.

9.2.2 Housing

Figure 9.8 shows a technical drawing of the housing for the MKII (Schmied, 2018). The diameter is reduced from 70 mm to 60 mm to allow better operation for users with smaller hands. The foot, made of plastic in this version, and the wooden top with the excitation pad are fixed with brass rings and screws, marked red in the drawing. This allows a smoother assembly and better access for maintenance. The acrylic tube was prone to shock, cracking easily. It has thus been replaced by one made of polycarbonate. Electronics and the battery are mounted on a median plane made of polycarbonate, which also holds the FSRs for the valve mechanisms.

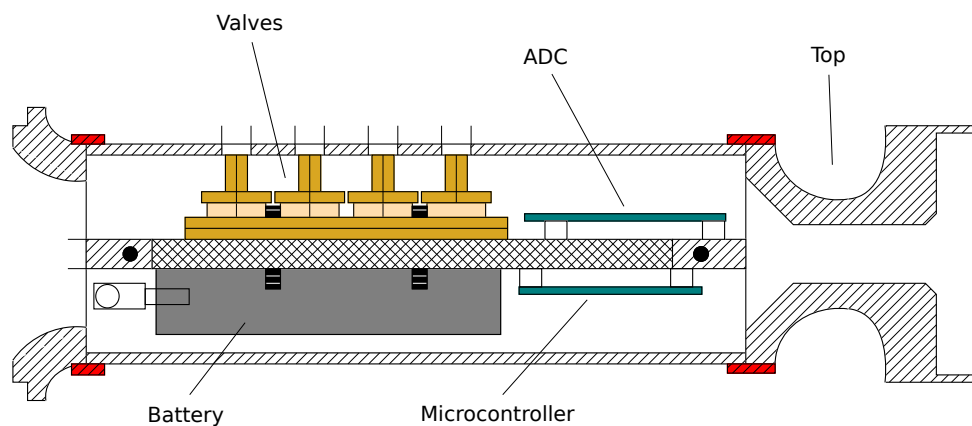


Figure 9.8: Longitudinal section of the BINBONG, side view (Schmied, 2018).

Figure 9.9 shows a technical drawing of the wooden top with the excitation pad. The four FSRs (Flex Sensing Resistors) are arranged in a square, allowing the detection of the position when applying force. The silicone cushions from the first version have been replaced, since the ones molded for this prototype fatigued after several hours of use. New inserts, colored beige in the drawings, are designed from felt, as used in instrument making. This material is more robust and still provides a soft action point.

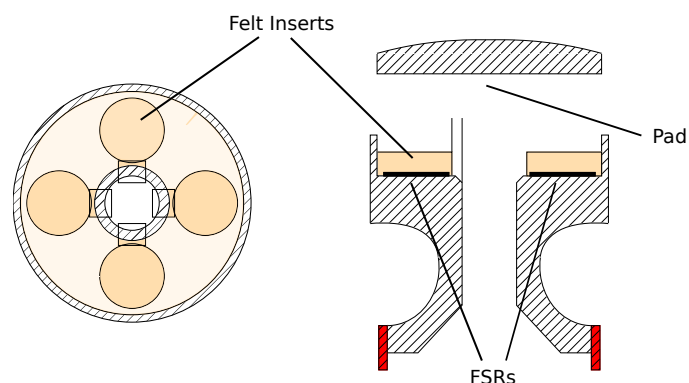


Figure 9.9: Wooden top with excitation pad, four FSRs and four felt inserts (Schmied, 2018).

9.2.3 Electronics

A *RedBear DUO* development board is used to process the sensor data and send control messages via WiFi. The board is based on an *ARM-Cortex-M3 STM32F205* micro controller, equipped with Bluetooth and WiFi communication. A dedicated software repository contains all files and libraries for the firmware (von Coler, 2020a).

Two custom printed circuit boards were designed by Anton Schmied and produced at TU Berlin for connecting the *RedBear DUO* with the additional components (Schmied, 2018). The analog-to-digital capacities of the board are extended with an *Adafruit ADS1015* 12-Bit 4 Channel ADC. An *Adafruit FLORA 9-DOF* inertial measurement unit was used in the first versions of the MKII for tracking the orientation. For the user study it was replaced by an *Adafruit* 9-DOF absolute orientation IMU with embedded sensor fusion (BNO055) for a more robust orientation tracking.

First prototypes of the MKII used a USB power bank, colored dark gray in Figure 9.8. Charging could thus be managed by the power bank and no additional circuitry was needed. However, this solution did not deliver steady power and caused outages of the device, especially when significantly discharged. In the makeover of the MKII, realized by Yrkkö Äkkijyrkkä at TU Berlin, it was replaced by a Lithium-ion battery with 3.7 V and 2500 mAh and an additional charging board, based on a *MCP73833* chip.

9.2.4 FSR Units

The BINBONG MKII has three force-based sensor units, namely the *Valves*, the *Pad*, and the *Ribbon*, as shown in Figure 9.10. Each unit is comprised of one or more force-sensing resistors. Individual sensor values of the pad, the valves and the ribbon are pre-processed before sending. Most control parameters are generated on the device itself, whereas others are calculated in the receiver software.

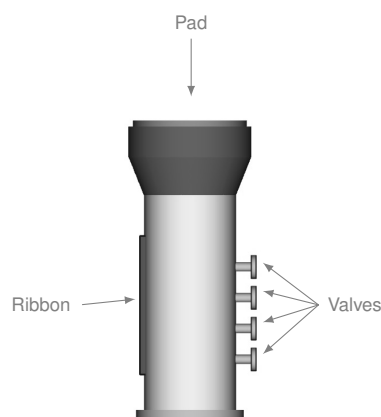


Figure 9.10: Force-based sensor units.

FSR Calibration

The eight FSRs of the valves and the pad are exposed to an initial tension through the design of the hardware. This is caused by the felt inlays between the four valve sensors and the mechanisms, respectively by the spring pulling the excitation pad towards the FSRs underneath it. This tension is individual for each sensor and is subject to changes. In order to compensate for this, each sensor is calibrated during the boot of the system by memorizing the tension f_{INI} at boot time. The measured force f_{FSR} is then normalized according to that value and the maximum value f_{MAX} for the respective sensor:

$$f_{FSR}^* = \frac{f_{FSR} - f_{INI}}{f_{MAX} - f_{INI}} \quad (9.1)$$

9.2.5 Valves

Force

The individual force applied to each of the four valves is sent to the receiver, which calculates the mean force on all valves. This mean force is scaled by the number of pressed valves. Hence, the full range can be used, independent of the activated valves.

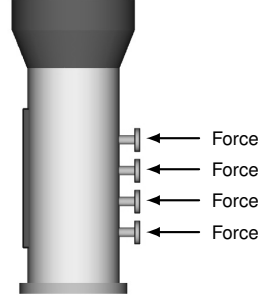


Figure 9.11: Four valves.

Pitch Mapping

The fundamental principle of the pitch mapping in this controller is a single-handed, binary pitch selection. Further, the pitch selection should only require minimal changes of the posture to allow swift transitions. Based on these demands, binary combinations of four fingers are mapped to one octave. Fingering systems similar to the one of the BINBONG are found in wind instruments and DMI controllers. Various fingering systems in digital musical instruments are based on instruments like the saxophone or trumpet (Hartvigsen, 2014, p. 6).

Binary combinations of four sensors are sufficient for selecting pitches within more than an octave. The case of no pressed sensor is discarded, leaving a total of 15 combinations:

$$N_{comb} = 2^4 - 1 = 15 \quad (9.2)$$

With an octave containing $N_{oct} = 12$ individual pitches, this allows a total of

$$\frac{N_{comb}!}{(N_{comb} - N_{oct})!} = 217945728000 \text{ permutations.} \quad (9.3)$$

Since the mapping in electronic musical instruments is not bound to any physical restrictions, as it is the case for most acoustical instruments, it can be freely chosen from these permutations. Hence, various strategies have been presented in the field of DMI research, as introduced in Section 4.4.3.

A simple mapping, based on binary counting, was chosen for the BINBONG at an early stage, due to its comprehensibility and explicitness. Binary counting is easily explained and follows a simple logic for sorting high and low values, allowing an easy orientation within the octave. Of course, people with experience in computer science and related fields are more likely to agree with these statements. As shown in Table 9.1, the least significant bit is assigned to the index (F2), whereas the most significant bit is assigned to the little finger (F5).

Table 9.1: Binary Code Mapping - *F5 = little finger, F4 = ring finger, F3 = middle finger, F2 = index* (von Coler, Treindl, Egermann, & Weinzierl, 2017).

F5	F4	F3	F2	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	C
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	C#
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	D
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	D #
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	E
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	F
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	F #
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	G
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	G #
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	A
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	A #
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	B
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	C'

9.2.6 Pad

Force

The BinBong sends all four FSR values to the receiver, where they are combined to one overall force value, independent of the location. This control parameter is intended to be used for the intensity, respectively the excitation.

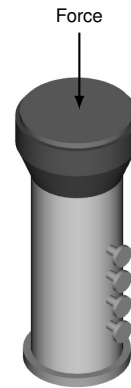


Figure 9.12: Force value of the pad.

Position

First experiments used the location of the pressure on two axis of the pad. In order to simplify the use of the interface, only the front-rear axis is used for the user experiment. The control value `position` is zero when the pad is pressed near the ribbon side and one when pressed on the valve side. This parameter is not generated on the device but in the receiver software.

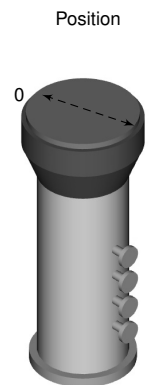


Figure 9.13: Position value of the pad.

9.2.7 Ribbon

The piezo-based octave selection of the first version was initially replaced with capacitive switches. Due to issues in the first user study with the MKII, it was replaced with a *Pololu* force-sensing linear potentiometer (Schmied, 2018, p. 9). This sensor unit generates three control parameters.

Position

The position of pressure is scaled from 0 - close to the foot, to 1 - close to the pad, across the full range of the ribbon. This parameter was initially intended for octave selection by dividing it into distinct areas. First tests showed that this is not easy to use.

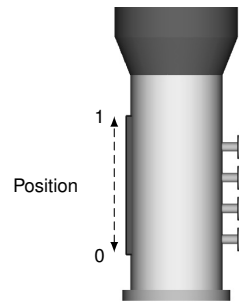


Figure 9.14: Position value from the ribbon sensor.

Force

The force applied when pressing the ribbon is used as a control parameter, independent of the position. This value is highly correlated with the force applied to the valves, which makes their individual use problematic.

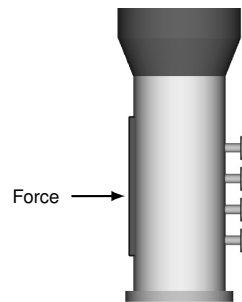


Figure 9.15: Force value from the ribbon sensor.

Swipe

For allowing a robust and intuitive octave selection, a swipe gesture is implemented on the device. The swipe parameter is an integer, which can be increased by one when swiping up the full length of the ribbon, and decreased by one through swiping down.

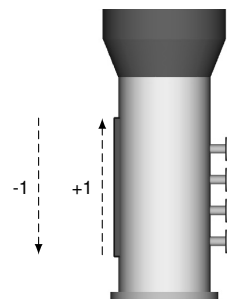


Figure 9.16: Swipe gesture of the ribbon controller.

9.2.8 IMU Parameters

The orientation of the device is tracked with Euler angles, respectively the principal axes pitch, roll and yaw, used in aviation. Figure 9.17 shows the BINBONG with its three axes of rotation. When holding the device, the valves point to the front. The x-axis then points from left to right, the y axis towards the front and the z-axis upwards. The three rotations around these axes are described in the following.

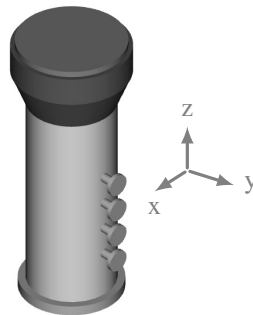


Figure 9.17: Three rotation axes of the BinBong.

Pitch

Pitch represents a rotation of the interface around the x-axis, as shown in Figure 9.18. It is limited to $\pm 90^\circ$. The pitch parameter is intended to control the vertical angle of incidence of a virtual sound source.

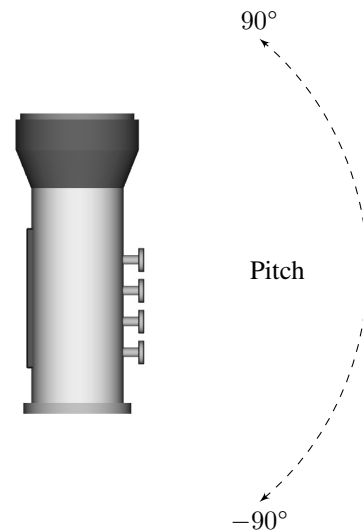


Figure 9.18: Rotation around the x-axis.

Roll

Roll represents a rotation of the interface around the y-axis, as shown in Figure 9.19. It is also limited to $\pm 90^\circ$. This IMU parameter has not been linked to a specific rendering parameter during the development process.

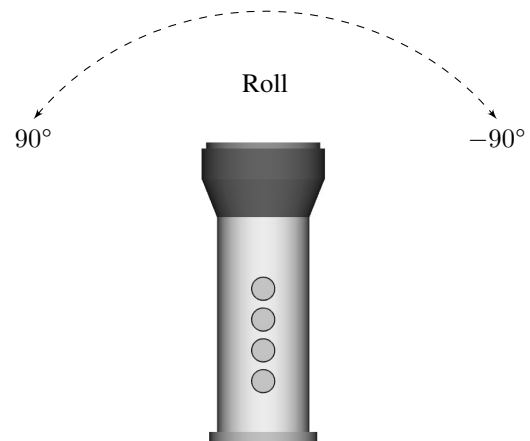


Figure 9.19: Rotation around the y-axis.

Yaw

Yaw represents a rotation of the interface around the z-axis, as shown in Figure 9.20. This control parameter was introduced for controlling the horizontal angle of incidence of the virtual sound source.

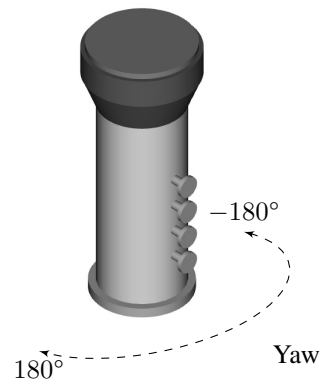


Figure 9.20: Rotation around the z-axis.

9.2.9 Output Values

In the recent implementation, the receiver software processes all parameters from the BINBONG at a sampling rate of 87 Hz, using the OSC protocol for transmission. Figure 9.21 shows all control parameters sent from the device to the receiver, grouped by sensor unit. The parameter *MIDI Note* is generated by both the valves and the excitation pad. In addition to the Euler angles, the raw acceleration values of the IMU are included. All control parameters, with the exception of the MIDI pitch, are normalized to values between 0 and 1. This allows a dynamic mapping to the synthesis parameters, as explained in Section 4.4.

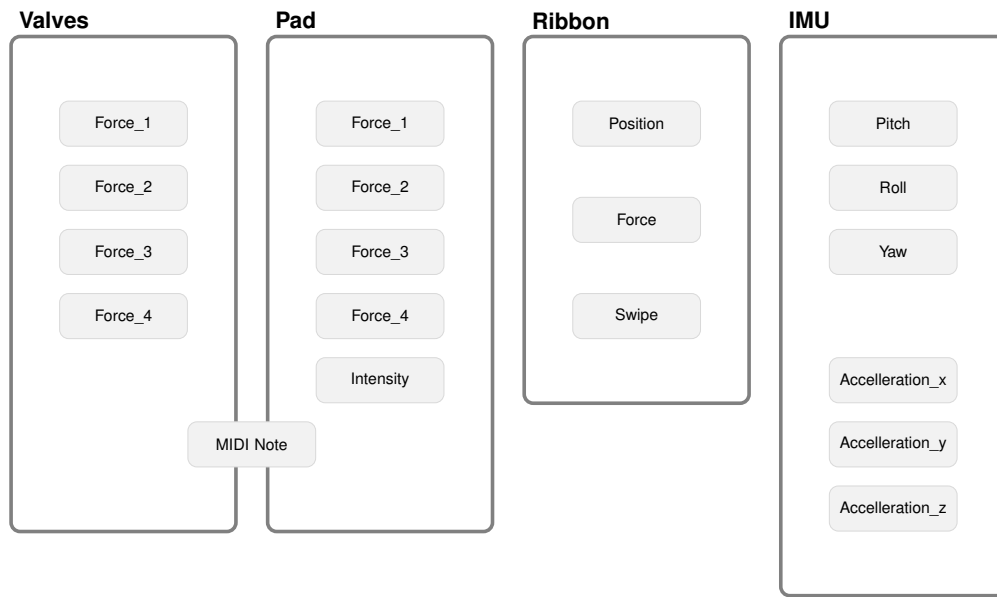


Figure 9.21: Control parameters sent from the BinBong.

Paths of all OSC messages start with `/bong/<IP_ADDRESS>/`, with `<IP_ADDRESS>` being replaced by the last octet of the device's IP address. Thus, multiple devices can be connected to one receiver and routed, individually. Tables 9.2 through 9.5 list all control parameters with their paths and arguments grouped by sensor units. Additionally, the device sends the calibration status of the IMU with the messages listed in Table 9.6.

Table 9.2: OSC messages for the valves.

OSC path	Arguments	Description
valve/1	[f]: 0 ... 1	Force applied to valve 1.
valve/2	[f]: 0 ... 1	Force applied to valve 2.
valve/3	[f]: 0 ... 1	Force applied to valve 3.
valve/4	[f]: 0 ... 1	Force applied to valve 4.
note	[f]: 0 ... 127 [f]: 0 ... 1	Pitch from valves with intensity.

Table 9.3: OSC messages for the pad.

OSC path	Arguments	Description
pad/1	[f]: 0...1	Force applied to pad sensor 1.
pad/2	[f]: 0...1	Force applied to pad sensor 2.
pad/3	[f]: 0...1	Force applied to pad sensor 3.
pad/4	[f]: 0...1	Force applied to pad sensor 4.
intensity	[f]: 0...1	Mean force applied to all pad sensors.

Table 9.4: OSC messages for ribbon controller.

OSC path	Arguments	Description
ribbon/position	[f]: 0...1	Position of touch on the ribbon.
ribbon/pressure	[f]: 0...1	Force applied to the ribbon.
ribbon/octave	[i]: -5...5	Octave selection (swipe up or down).

Table 9.5: OSC messages for acceleration and orientation.

OSC path	Arguments	Description
imu/accel/x	[f]: 0...1	Acceleration along the x-axis.
imu/accel/y	[f]: 0...1	Acceleration along the y-axis.
imu/accel/z	[f]: -5...5	Acceleration along the z-axis.
orientation/pitch	[f]: -90...90	Rotation around the x-axis.
orientation/roll	[f]: -90...90	Rotation around the y-axis.
orientation/yaw	[f]: -180...180	Rotation around the z-axis.

Table 9.6: OSC messages for IMU calibration.

OSC path	Arguments	Description
cal/sys	[i]: 0...3	System calibration status.
cal/gyro	[i]: 0...3	Gyroscope calibration status.
cal/accel	[i]: 0...3	Accelerometer calibration status.
cal/mag	[i]: 0...3	Magnetic calibration status.

9.3 Receiver & Mapping Software

9.3.1 Mapping in Spatial Sound Synthesis

In the case of spatial sound synthesis, mapping between control parameters, synthesis and spatialization parameters happens in an integrated way. The standard model of the digital musical instrument, as introduced in Section 4.4.1, can be extended to include the spatial aspect. Figure 9.22 shows this extended model, in which the mapping distributes the control parameters. Audio is sent from the synthesis process to the spatialization. This allows the joint control of timbral and spatial properties which is the major strength of spectro-spatial sound synthesis.

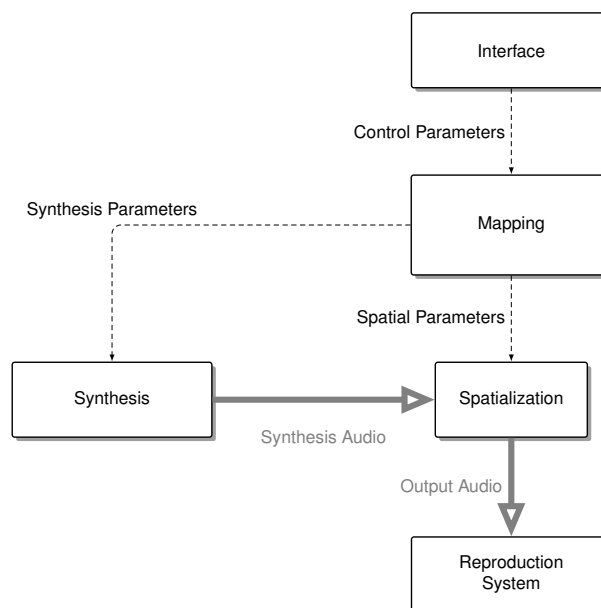


Figure 9.22: The extended DMI model for spatial sound synthesis.

9.3.2 BINBONG Receiver

All control parameters from the BINBONG are sent to a Linux audio server, running the GLOOO synthesis system and a spatial rendering software. Pure Data is used for receiving and processing the data from the control device and distributing it to synthesis and spatialization. This ensures maximum flexibility when exploring different mappings or processing steps for the control parameters.

In addition, the Pure Data patches allow a basic control of the synthesis engine without an input device, using basic GUI elements. For test and demo purposes, the playback of previously prepared control trajectories for a re-synthesis of recorded phrases is also included. The collection of patches is not only a development tool or a prototype for a user interface. For experienced users it also resembles the most efficient and powerful way of working with the full system.

The main receiver patch `BINBONG.pd`, shown in Figure 9.23, is part of the BINBONG’s software repository (von Coler, 2020a). It visualizes the main interface parameters through vertical sliders, grouped by sensor unit. The individual ID of the interface to be received is passed to the patch as a creation argument. Within the receiver patch, additional sensor fusion is performed for generating the pad position parameter and the combined valve pressure. All resulting control parameters are provided by the patch as global variables in Pure Data.

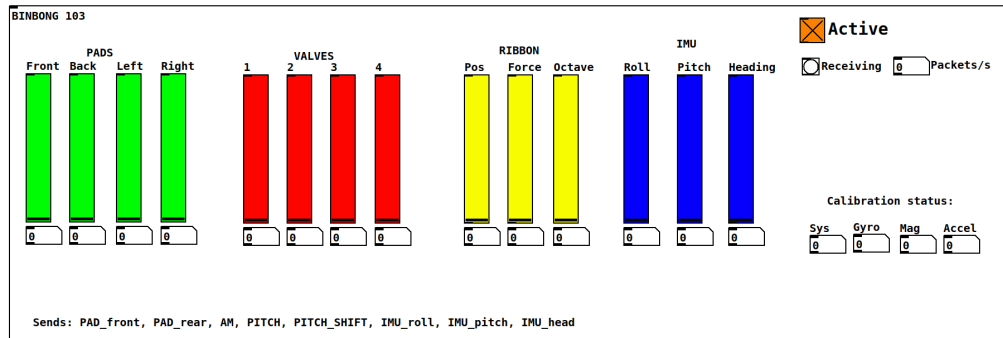


Figure 9.23: BinBong receiver patch.

9.3.3 Spatial Distribution

Depending on the spectral dispersion mode, the synthesis engine provides up to 104 outputs, which are routed to dedicated point sources in the spatial rendering software via Jack. The spatial distribution of these point sources, respectively their individual positions, can be controlled during performance. Together with the spectral dispersion, introduced in Section 8.2, this results in the *spectro-spatial dispersion* of the synthesis system.

A manageable set of metaparameters needs to be defined for controlling the spatial distribution of this large number of point sources. For the user study in the following chapter, the model of a virtual sound source with a position in space and a spatial extent, shown in Figure 9.24, is introduced. The source position is controlled via two angles, namely the azimuth, α and the elevation, ϵ , as well as the distance d to the source center. The extent of the virtual sound source is controlled with a single parameter S , the spread. This simple model, with only one parameter for the spatial extent, is the reduction of the more detailed models from the research on virtual acoustics and the theory of electroacoustic music, presented in Section 4.3.2. Informal tests with the complete system indicated that width, height and depth are not distinguishable in a more detailed model. Moreover, the virtual sound source would be harder to control with additional parameters.

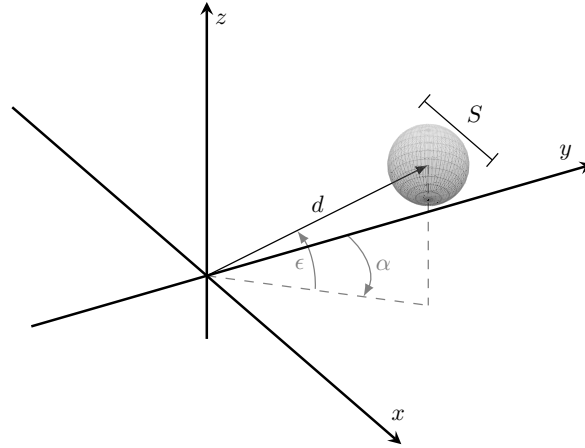


Figure 9.24: Virtual sound source with a position and spatial extent.

The parameters α , ϵ , d and S of the virtual sound source are presented to the user as parameters of direct spatial control. The wrapper patch `spatial-mapping.pd` holds an instance of the abstraction `partial-posotion.pd` for each point source, calculating its individual position, depending on these metaparameters. Individual positional parameters α_i , ϵ_i and d_i of the source with index i are derived. During development and for the experiment, a spherical shape is approximated. The scaling factors have been tuned heuristically to result in a quasi homogeneous configuration:

$$\alpha_i = \alpha S \cdot (d + 1) \cdot ((i \bmod 2)) - 0.5 \cdot 2 \sin\left(\frac{i}{2}\right) \quad (9.4)$$

$$\epsilon_i = \epsilon S \cdot (d + 1)^{-1} \cdot ((i \bmod 2)) - 0.5 \cdot 0.5 \cos\left(\frac{i}{12}\right) \quad (9.5)$$

$$d_i = 10 d + 3 S \cdot ((i \bmod 2)) - 0.5 \cdot \sin\left(\frac{i}{4}\right) \quad (9.6)$$

Individual parser patches, like `osc_to_sc_spat.pd` in Figure 9.25, route the above calculated individual point source positions to the according rendering software. To this point, Panoramix and the custom SuperCollider tool `SC_SPAT` (von Coler, 2020f) are implemented.

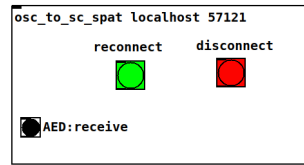


Figure 9.25: Patch for sending OSC to the spatial rendering software.

9.3.4 Synthesis Control

The `glooo_param.pd` patch, shown in Figure 9.26, is part of the repository for the synthesis software (von Coler, 2020c). It controls global parameters of the synthesis engine, such as volumes, synthesis modes, partial activity and pitch shift. All relevant parameters are introduced and listed in Section 8.3.6.

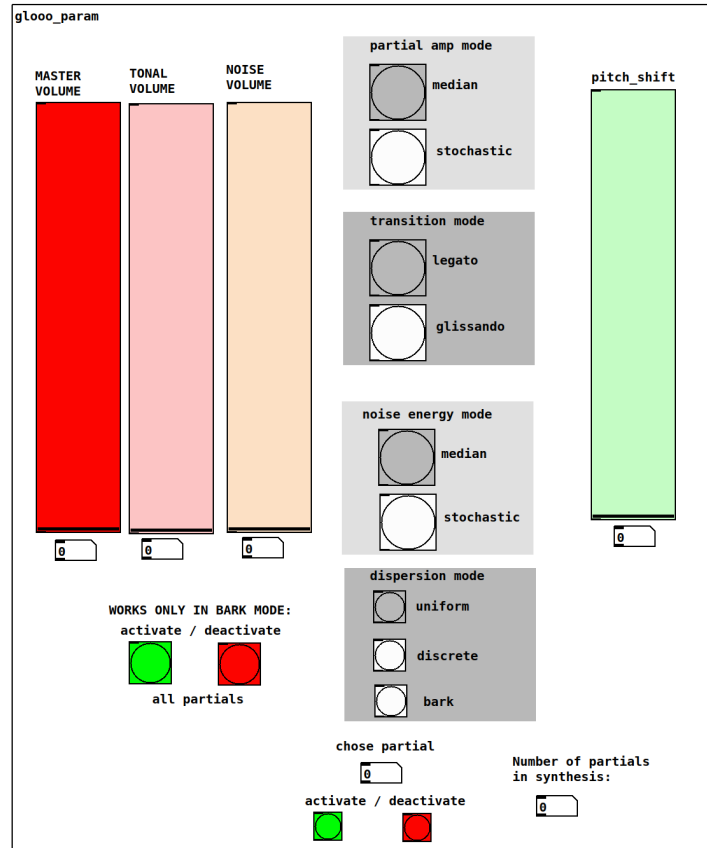


Figure 9.26: GLOOO parameter settings patch.

The repository features additional means for a direct control of the synthesis system with GUI sliders for pitch and intensity in the patch `GL000_test.pd`. Further, this test patch includes generators for automated parameter modulations, used for testing and debugging the synthesizer. An abstraction `osc_to_synth.pd`, shown in Figure 9.27, gathers the global control parameters from the different patches and sends them to an instance of the synthesis engine, specified by an IP address and a UDP port as creation arguments.

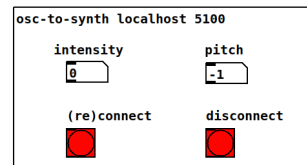


Figure 9.27: Patch for sending OSC to the synthesis software.

9.3.5 Mapping Environment

With control parameters, synthesis parameters and the metaparameters for spatial distribution at hand, the actual mapping is the final step before the instrument can be used. Initially, mappings were *hard-patched* in abstractions, allowing only cumbersome changes. Only for the concluding study on user defined mappings, the idea of a graphical mapping environment came up. Figure 9.28 shows the fully patched environment, realized with standard Pure Data GUI elements. For the sake of explicitness in the user study, the mapping environment allows only one to many connections and does not process one-to-many connections.

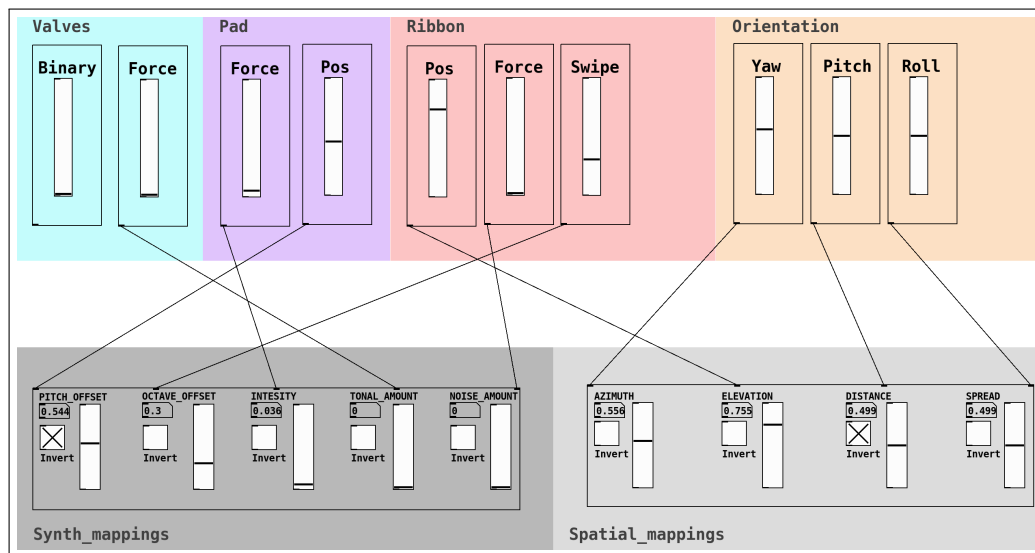


Figure 9.28: Mapping GUI as used in the user study, fully patched.

Control parameters are arranged in the upper half, grouped by the control units *Valves*, *Pad*, *Ribbon* and *Orientation*, in colored boxes. The instantaneous values of all control parameters are visualized by vertical sliders. The only hard-wired connection is between the binary values of the valves and the pitch. Rendering parameters are located in the lower half, divided into synthesis parameters and spatial parameters by gray boxes. The instantaneous values of all rendering parameters are presented by vertical sliders and number boxes. Each rendering parameter can be inverted using a toggle switch. For Pure Data users, mapping is straightforward. For others, it is easy to acquire. Connections between control and rendering parameters can be established by drag and drop. Existing connections can be deleted by selecting and pressing the delete key. All parameters are normalized for allowing all possible connections without exceeding parameter limits.

Part III

User Study & Conclusion

User Study

A user study is conducted with the full system, focusing on user-defined mappings. Background and motivation for the chosen approach are discussed in Section 10.1. Section 10.2 explains the full experimental setup, including the loudspeaker system. The method and the included sub-tasks are introduced in Section 10.3. Results and discussion are presented in Section 10.4, respectively Section 10.5.

10.1 Background

Since the expressive control of spatial sound synthesis with a dedicated interface is still a novel approach, it is of special interest how users experience the system as a whole. From a usability point of view, participants could be provided with the full instrument to rate specific aspects. Initial plans were based on the evaluation of pre-defined mappings and spatialization modes through a task/performance paradigm. Similar approaches were used for the evaluation of the first version of the BINBONG (von Coler et al., 2017), an unpublished usability study of the second version and in a transition modeling experiment (von Coler, Götz, & Lepa, 2018). One finding of these experiments was that this approach did not invite participants to explore the systems in an enjoyable way. In the past years, the NIME community has already shifted from such task based usability paradigms towards a more experiential focus (Brown, Nash, & Mitchell, 2017). This is necessary for evaluating the hedonic qualities and the joy of use, rather than pragmatic qualities, as in generic product design.

With all components at hand, the mapping between interface, sound synthesis, and spatialization is the final crucial step for turning the system into a musical instrument. Since the BINBONG was developed specifically for the spatial sound synthesis system, its control parameters were already intended for specific purposes. Still, different mappings were considered and explored during the development. As a result, the graphical user interface for parameter mapping, explained in Section 9.3.5, was conceived during this phase.

This study was first intended as a final step of user-driven design for finding the best mappings between interface, synthesis engine and spatialization. The most promising mappings could then be implemented on the device. However, the mapping GUI offers way more opportunities. Mapping decisions of the participants can help in the evaluation of specific components of the system. Most importantly, the system is turned into a flexible, versatile instrument and research tool. This tool can be used to investigate the concept of user-defined mapping in the context of spatial sound synthesis. Through detailed logging of control data and mapping process it is possible to explore mapping strategies of participants.

10.2 Test Setup

10.2.1 Space and Equipment

The user study was conducted in the sound field synthesis studio at TU Berlin, located at Einsteinufer 17c. Figure 10.1 shows room EN 325, the *small studio*, as configured for the user study. Besides the dome for Ambisonics rendering, a WFS system and a ring of eight speakers are installed, but not used in this experiment. Large studio tables were removed, creating an adequate area for free movement in the center of the room. A square of 1 m² was marked on the floor with white tape, representing the *sweet area*. A table with computer screen, mouse and keyboard was placed next to the square, allowing participants to work with the mapping environment from Subsection 9.3.5 and to control the test procedure. A second table at the far right side was used for paperwork and surveys.

The test management was located in the next-door studio EN 324. A talk back system allowed the communication between both rooms. The mapping screen in the experimental space was mirrored to the test management. This was necessary to ensure that participants did not experience problems with the mapping GUI in Pure Data or deviate from the prescribed procedure.

10.2.2 The Runtime System

Rendering Software

The output of the GLOOO synthesis system can, in combination with the Pure Data mapping software, be used with any spatial rendering software which is capable of receiving OSC messages for controlling the virtual acoustic scene. Before integrating the custom *SC_SPAT* solution into the project, IRCAM's *PanoramixApp* (Carpentier, 2016) was used throughout the development. For the user study, version 1.3.3 (build Oct 4 2018) was installed on the system.

Loudspeaker Setup

Among other systems, the studio EN 325 is equipped with a dome of 21 *Genelec 8020* speakers and two subwoofers, intended for Ambisonics rendering. The configuration of the loudspeakers



Figure 10.1: Test setup in the small studio (EN 325) at TU Berlin.

is shown in Figure 10.2. It consists of four levels. The two lowest levels feature eight loudspeakers each, at a height of approximately 0.8 m, respectively 1.7 m. A square of four loudspeakers is located at 2.50 m height above the sweet area and a central top speaker at 3.1 m height.

Rendering Hardware

The Linux audio server used in the study is based on an *Asus PRIME B350-PLUS* motherboard with an *AMD Ryzen 7 1800X* eight-core processor, at a maximum frequency of 3.6 GHz with 16 virtual CPUs. A total of 16 GB RAM with two *Kingston KHX3200C16D4/8GX* 8 GB modules is installed. An *RME HDSPe MADI* audio interface connects the machine to the digital-to-analog converters of the studio setup. The renderer is running a *Ubuntu Studio 18.04*.

A JACK server was started with a sampling rate of 48 kHz and a buffer size of 256 samples. These settings result in a rather high latency. However, with the software used in the experiment, the simultaneous control of a large number of sound sources did not allow lower buffer sizes. For the violin synthesis and the spatial control, the resulting latency is within an acceptable range. A *FRITZ!Box 4040* was used for connecting the BINBONG to the rendering computer.

Data Recording

A custom Python-based software (von Coler, 2020e) is used to record all data from the control and mapping procedure as continuous streams. It is running on the rendering machine, receiving all parameters from the Pure Data receiver and mapping software at a fixed sampling rate. This includes all control parameters from the BINBONG, as well as the complete state of the mapping.

Data is stored in several dedicated text files with UNIX Epoch time markers.

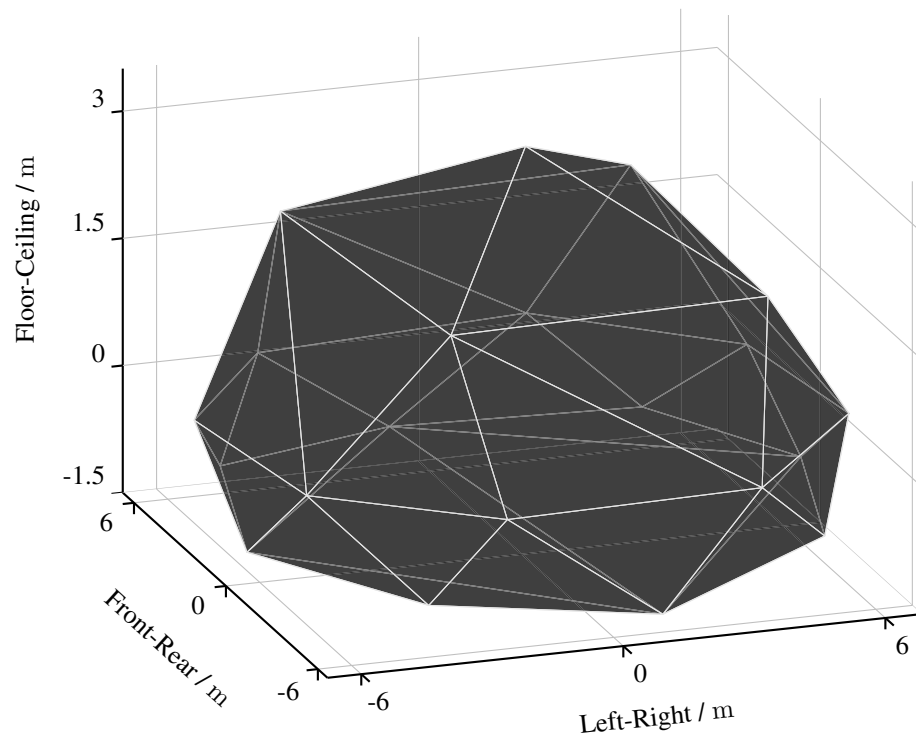


Figure 10.2: Geometry of the dome with 21 loudspeakers in the TU Studio (EN 325).

10.3 Method

The experiment consists of two consecutive parts, namely the mapping stage and the gesture stage. During the mapping stage, participants create their individual patches using the graphical user interface. In the following gesture stage, participants are asked to perform gestures, based on simple verbal descriptions, using their custom mapping. Before starting the mapping stage, users were thoroughly introduced to the interface and the synthesis system, using a printed user manual. This introduction lasted between 15 and 20 minutes.

10.3.1 Mapping Stage

During the mapping stage, participants were given 30 minutes to create an individual mapping, using the Pure Data patch interface introduced in Section 9.3.5. It allows the connection between ten *control parameters* from the interface and nine *rendering parameters*, consisting of synthesis parameters and spatial parameters. All parameters are listed in Table 10.1. Control parameters are introduced in Section 9.2. Synthesis parameters and spatial parameters are explained in detail in Section 8.3.6, respectively Section 9.3.3. Each parameter was explained to the participants in the introduction.

Table 10.1: Control parameters and rendering parameters in the user study.

Control Parameters	Rendering Parameters
Valves: Binary	Pitch Offset
Valves: Force	Octave Offset
Pad: Force	Intensity <i>Synthesis</i>
Pad: Position	Tonal Amount
Ribbon: Position	Noise Amount
Ribbon: Force	Azimuth
Ribbon: Swipe	Elevation <i>Spatial</i>
IMU: Pitch	Distance
IMU: Roll	Spread
IMU: Yaw	

Mapping Objective

After being introduced to the system, participants were presented the following text, outlining the mapping objective:

The objective of this part is to create an enjoyable mapping, which offers the most expressive control over all synthesis and spatialization parameters.

This target should animate the participants to explore the system and the mapping in a playful way, without creating performance pressure. They were additionally informed that their mapping

is not expected to result in a natural or plausible sound. Only their personal preference is the decisive factor.

Mapping Directives

In order to generate comparable results, four mapping directives were presented to the participants, listed in Table 10.2. These directives allow one-to-many mappings and exclude many-to-one mappings. Abbreviated, this can be put into one single demand: Every rendering parameter must have exactly one connection after completing the mapping stage.

Table 10.2: The four mapping directives.

- 1. Every parameter of synthesis and spatialization must be influenced through the mapping.*
- 2. Control (interface) parameters may remain unconnected.*
- 3. A single control parameter may be mapped to multiple synthesizer or spatialization parameters.*
- 4. A synthesis or spatialization parameter must not have more than one control parameter connected to its input.*

Procedure

After being introduced to the task, the participants were left in the studio on their own. They were given 30 minutes for completing the mapping, without presenting a clock in the studio. Participants could notify the test management through the talk back system if they were satisfied with their mapping before the granted time expired. Test management informed the participants after 30 minutes that they need to finish their mapping, leaving another five minutes for completion and last changes.

Mapping Survey

After finishing the mapping task, participants completed an intermediate survey, referring to their procedure in the mapping stage. This survey contains two prompts, listed in Table 10.3. Responses were given in writing, each on a full DIN A4 page.

Table 10.3: Questions in the mapping survey.

- 1. Explain your procedure during the mapping process.*
- 2. Explain your choice for the final mapping.*

10.3.2 Gesture Stage

The gesture stage represents a hybrid between free play and task-based evaluation. Participants are asked to perform gestures, defined by nonspecific verbal descriptions. Each of these gestures, listed in Table 10.4, refers to a specific attribute of the rendering system. The tasks do not specify exact gestures but encourage the development of individual gestures. By this procedure, participants are encouraged to experience the different aspects of the instrument and their own mapping, without constraining them through specific tasks.

Table 10.4: Single instructions for the gesture stage.

Gesture	Description
Spatial 1	<i>Perform a gesture or a sequence of gestures resulting in a full rotation of the sound in the horizontal plane, respectively the azimuth.</i>
Spatial 2	<i>Perform a gesture or a sequence of gestures focusing on an increase and/or decrease of the sound's elevation.</i>
Spatial 3	<i>Perform a gesture or a sequence of gestures focusing on an increase and/or decrease of the source distance.</i>
Spatial 4	<i>Perform a gesture or a sequence of gestures focusing on an increase and/or decrease of the source spread.</i>
Pitch	<i>Perform a gesture or a sequence of gestures focusing on the modulation of the pitch.</i>
Intensity	<i>Perform a gesture or a sequence of gestures which focus on the modulation of the intensity.</i>
Tonal-Noise Ratio	<i>Perform a gesture or a sequence of gestures focusing on the modulation of the tonal to noise ratio.</i>
Arbitrary (1-3)	<i>Perform a gesture or a sequence of gestures at will.</i>

Procedure

Participants were introduced to the gestures in detail, explaining every item in Table 10.4. Afterwards, they were left in the studio on their own with an indefinite amount of time for completing all ten subtasks. They were asked to complete them in the order shown in Table 10.4. A copy of the printed instructions remained in the studio. Each subtask could be developed and rehearsed as long as desired. Once settled for a gesture, participants had to record one single trial using the Pure Data control interface shown in Figure 10.3. This recording could not be repeated. The mapping GUI from the previous stage was hidden during the gesture part.

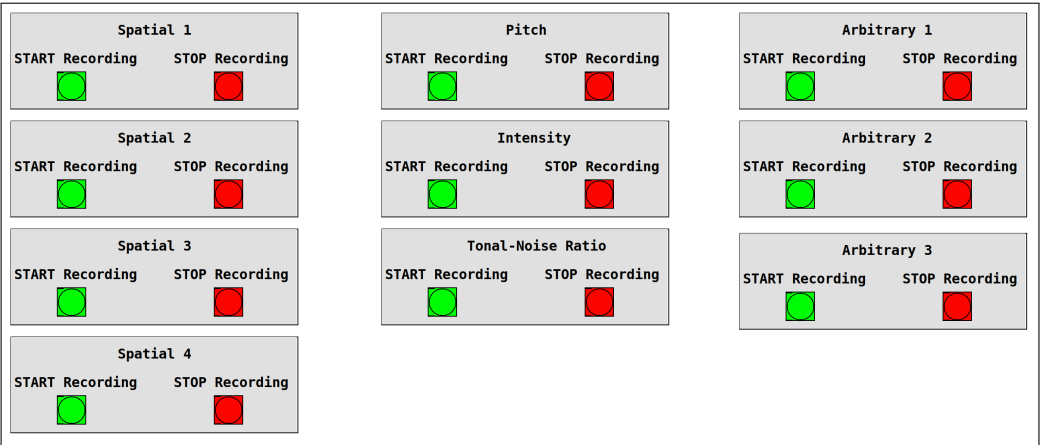


Figure 10.3: Control interface for the gesture stage.

Gesture Survey

After recording all ten gestures, participants were handed out a survey with 12 statements, listed in Table 10.5. These statements refer to attributes of the full rendering system and are used to asses the satisfaction of users with the system after performing the gestures. Responses for each attribute were given on a Likert scale with seven balanced responses. Items ranged from *Completely Disagree* to *Completely Agree*, as shown in Figure 10.4. Scales were completed by the participants with pen and paper.

Table 10.5: 12 items in the gesture survey.

- 1. The note selection could be controlled satisfactorily.
- 2. The octave selection could be controlled satisfactorily.
- 3. The pitch deviation could be controlled satisfactorily.
- 4. The intensity could be controlled satisfactorily.
- 5. The tonal level could be controlled satisfactorily.
- 6. The noise level could be controlled satisfactorily.
- 7. The tonal to noise ratio could be controlled satisfactorily.
- 8. The position of the sound could be controlled satisfactorily.
- 9. The azimuth of the sound could be controlled satisfactorily.
- 10. The elevation of the sound could be controlled satisfactorily.
- 11. The distance could be controlled satisfactorily.
- 12. The source width could be controlled satisfactorily.

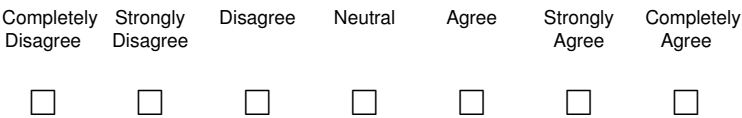


Figure 10.4: 7 point Likert scale for the gesture survey.

10.3.3 Additional Surveys

After completing both stages, additional surveys were handed out to the participants. A custom survey features two statements on the expertise in relation to the synthesis system, listed in Table 10.6. Three general questions are asked on the use of the full system, listed in Table 10.7. Responses to all items were given on the seven point Likert scales shown in Figure 10.4 with pen and paper.

Table 10.6: Experience-related statements in the survey.

- *You have experience with spatial audio.*
- *You have experience with electronic musical instruments.*

Table 10.7: General feedback on the full system.

- *Do you have any comments on the interface in combination with the synthesis and spatialization?*
- *Which aspects of the complete system did you enjoy?*
- *Which aspects of the complete system were unpleasant?*

Two additional validated surveys are completed by the participants. The full self-report questionnaire from the GMSI, the *Goldsmiths Musical Sophistication Index* (Müllensiefen, Gingras, Musil, & Stewart, 2014), comprises 38 items, covering five aspects of musical expertise, namely *Active Engagement*, *Perceptual Abilities*, *Musical Training*, *Singing Abilities* and *Emotions*. The *General Musical Sophistication* factor combines all five categories. Scales with seven points are used for all 38 items. Participants completed the survey with pen and paper.

The UEQ, the *User Experience Questionnaire* (Laugwitz, Held, & Schrepp, 2008) measures the user experience through 26 items on seven point rating scales. The six scales *Attractiveness*, *Perspicuity*, *Efficiency*, *Dependability*, *Stimulation* and *Novelty* are generated from the individual items. Based on these scales, the groups *Pragmatic Quality* and *Hedonic Quality* are created. The UEQ is designed for product evaluation and thus refers to *The Product*. For the application in this study, the term product was replaced with the term *system*. Ratings thus relate to the use of the complete system, including interface, mapping, synthesis and sound reproduction. The questionnaire was completed by the participants with pen and paper.

10.4 Results

10.4.1 Sample

A total of 22 persons were invited to the study, recruited through mailing lists of the Audio Communication Group. Four participants were declared test runs, since the procedure and setup were refined, afterwards, leaving 18 tests for evaluation. The recording of participant 06 does not start with an empty patch, as a thorough investigation of the mapping procedure revealed. The mapping stage was either started with previously patched connections or the recording was started too late. Since this can not be reconstructed and has an influence on the results of the mapping procedure, the participant is excluded from all evaluations. This leaves 17 valid data sets.

The final sample includes 14 male and three female participants with a mean age of 28.5 years ($SD = 8.2y$). Two persons were left-handed and 15 right handed. The sample includes ten master students, one Phd student and three research associates from the Audio Communication Group. In addition, one freelancing composer, one freelancing sound designer and a software developer and alumnus of the Audio Communication Group took part in the study.

Responses to the 7-point likert scale on the experience with spatial audio, ranging from *completely disagree* (0) to *completely agree* (6) result in a mean experience of 4.65 ($SD = 1.6$) which is rounded to a *strongly agree*. Responses to the 7-point Likert scale on the experience with electronic musical instruments, ranging from *completely disagree* (0) to *completely agree* (6) result in a mean experience of 4.65 ($SD = 1.28$) which is rounded to a *strongly agree*. Figure 10.5 shows the mean experience from these values with the individual answers.

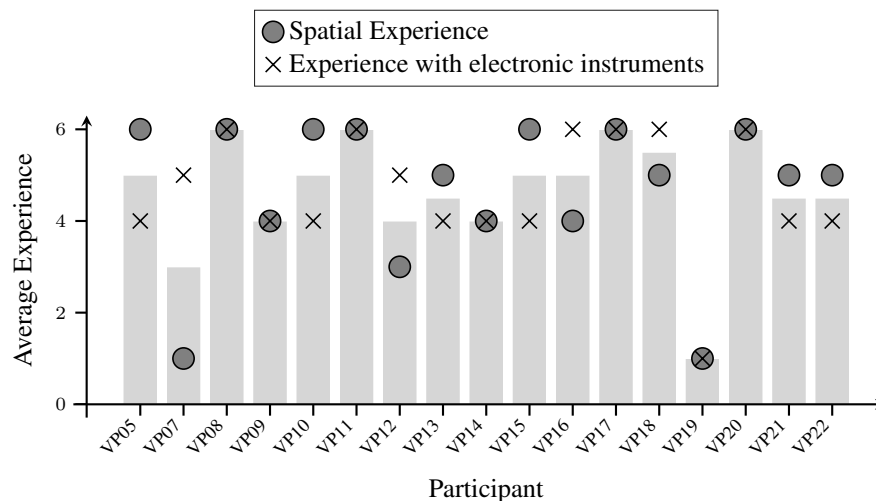


Figure 10.5: Average experience of participants from self-assessment.

10.4.2 GMSI

Figure 10.6 shows the ratings of all participants for the six major scales of the GMSI self-assessment survey. The sample has mean values of $\mu = 46.23$ for *Active Engagement* (SD = 9.76), $\mu = 51.06$ for *Perceptual Abilities* (SD = 6.73), $\mu = 34.47$ for *Musical Training* (SD = 8.49), $\mu = 36.29$ for *Emotions* (SD = 4.00), $\mu = 32.41$ for *Singing Abilities* (SD = 9.05) and $\mu = 89.41$ for the *General Sophistication* (SD = 15.67).

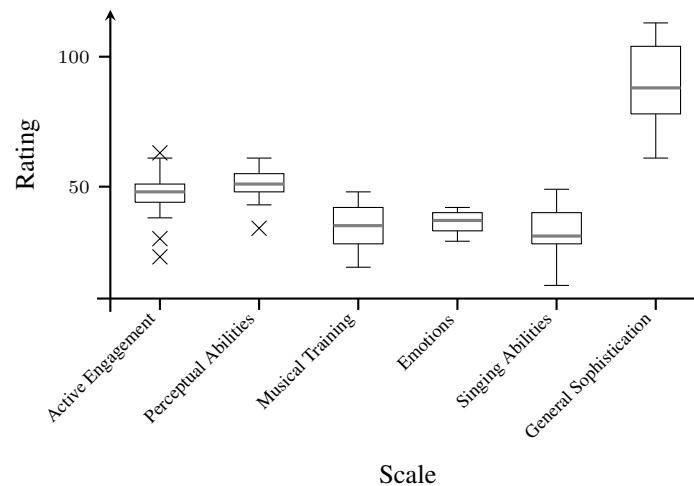


Figure 10.6: GMSI results.

Table 10.8 shows the statistics for 147,633 participants gathered for the development of the GMSI. The sample can be considered a cross section of the population in mainly Western and English-speaking countries, gathered in a large scale online survey (Müllensiefen, Gingras, Musil, & Stewart, 2014). Figure 10.7 shows the individual scores in the General Sophistication for each participant.

Table 10.8: Statistics of 147,633 participants from the GMSI (Müllensiefen, Gingras, Musil, & Stewart, 2014).

	Active En- gagement	Perceptual Abilities	Musical Training	Emotions	Singing Abilities	General So- phistication
Min.	9	9	7	7	6	18
Max.	63	63	49	49	42	126
Mean	41.52	50.20	26.52	31.67	34.66	81.58
SD	10.36	7.86	11.44	8.72	5.04	20.62

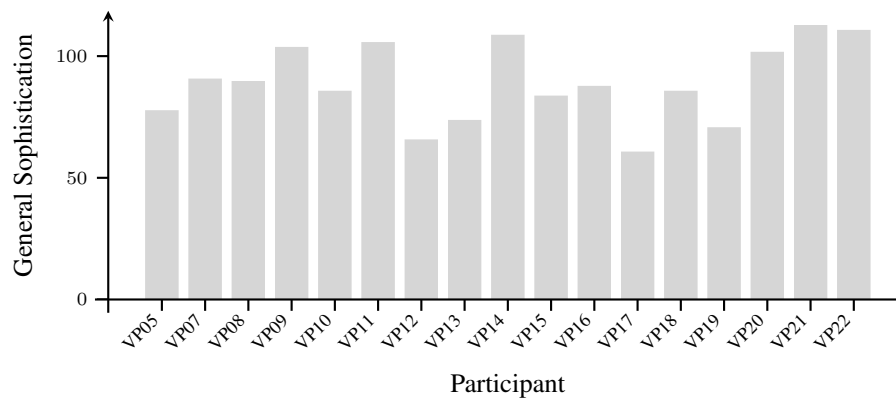


Figure 10.7: General Sophistication index for the individual participants.

10.4.3 UEQ

Mean Ratings

Mean ratings and variances across all participants for the major scales from the UEQ are listed in Table 10.9. The UEQ offers a benchmark test for comparing the results to an existing data set with ratings from 20190 persons and 452 studies. The set includes responses to business applications, development tools, web shops or services, social networks, mobile applications, and other products.

Table 10.9: Mean values and variances for the major scales of the UEQ.

	Attractiveness	Perspicuity	Efficiency	Dependability	Stimulation	Novelty
μ	1.314	0.706	0.926	0.941	1.691	1.912
σ^2	0.700	1.778	1.098	0.949	0.513	0.269

Figure 10.8 shows the ratings for the the synthesis system compared to the benchmark. It performs *above average* for Attractiveness (25 % of results better, 50 % of results worse), *bad* in Perspicuity (25 % worst results), *below average* in Efficiency and Dependability (50 % of results better, 25 % of results worse), *good* in Stimulation (10 % of results better, 75 % of results worse) and *excellent* in Novelty (in the range of the 10 % best results).

Results of the UEQ can be further aggregated resulting in the *pragmatic quality* (Perspicuity, Efficiency, Dependability) and the *hedonic quality* (Stimulation, Originality). Scores of the system for the three main qualities are listed in Table 10.10.

Table 10.10: Three main qualities of the system in the UEQ.

Attractiveness	1.31
Pragmatic Quality	0.86
Hedonic Quality	1.80

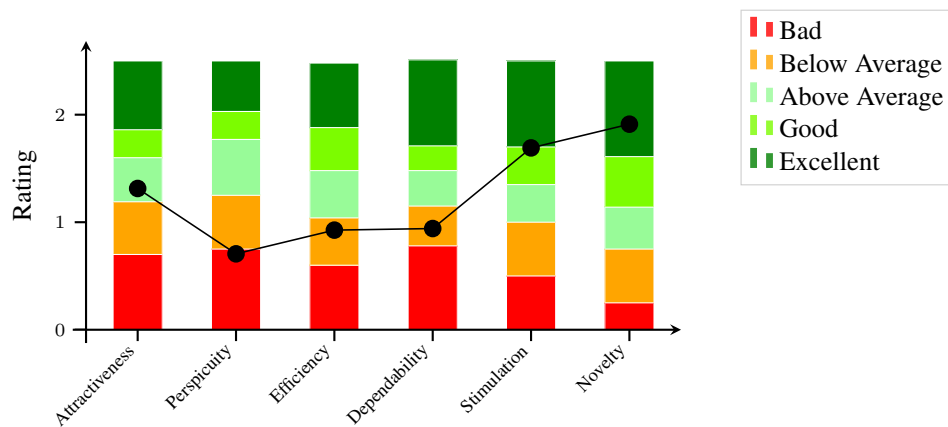


Figure 10.8: UEQ benchmark results.

10.4.4 Gesture Survey

Responses from the 12 rating scales for the satisfaction of controlling the synthesis system are mapped to values between -3 for *completely disagree* and $+3$ for *completely agree*. Figure 10.9 shows a box plot for the responses of all participants, grouped by attribute. The results for the individual participants, to all attributes are shown in Figure 10.10.

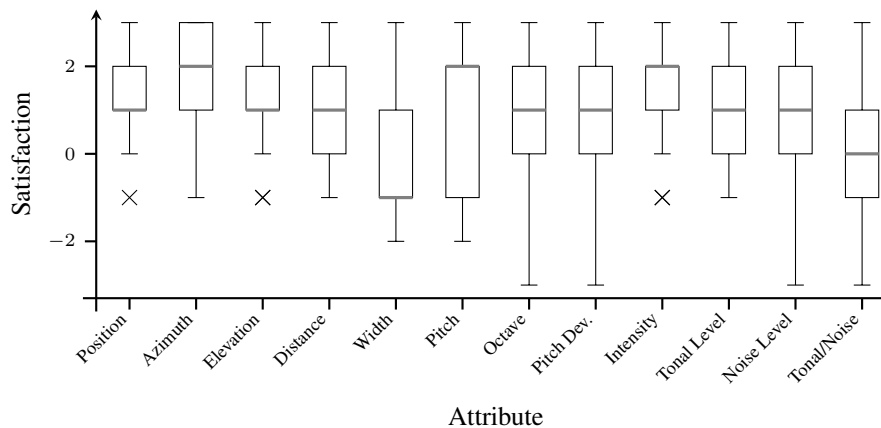


Figure 10.9: Box plots with ratings for each attribute.

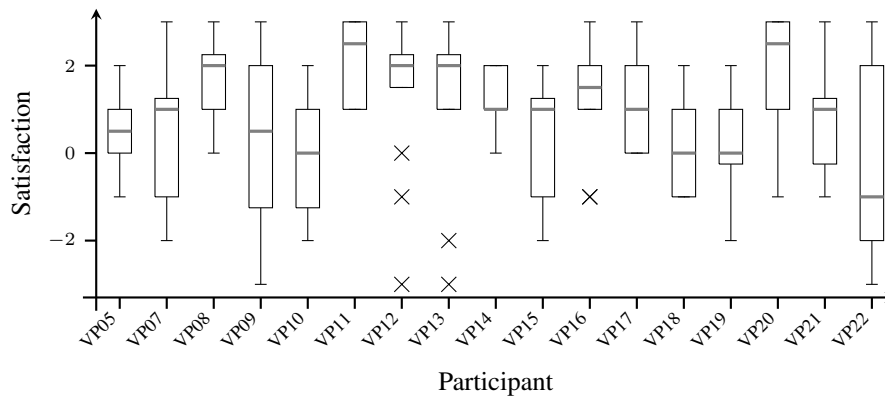


Figure 10.10: Box plots with ratings from each participant for all 12 attributes of the survey.

A k-means clustering of the participants is performed, using the 12 rating scales and two clusters. The resulting clusters have a size of eight, respectively 9 participants. A 3x12 repeated measure analysis of variance (ANOVA) is calculated with the between variable *Cluster* (0,1) and the within variable *Attribute* (Position, Azimuth, Elevation, Distance, Width, Pitch, Octave, Pitch Dev, Intensity, Tonal Level, Noise Level, Tonal/Noise) for the dependent variable *Rating*. Calculations were performed with the function `mixed_anova()` from the *Pingouin* package in Python. Results show a significant influence on the *Rating* only for the variable *Cluster* ($p < 0.001$), not for *Attribute* or the interaction. A comparison of mean ratings results in the assignment of participants to groups presented in Table 10.11.

Table 10.11: Two groups of different average satisfaction.

Satisfaction: High	VP08	VP11	VP12	VP13	VP14	VP16	VP17	VP20	
Satisfaction: Low	VP05	VP07	VP09	VP10	VP15	VP18	VP19	VP21	VP22

10.4.5 Mapping Process

Given a soft limit of 30 minutes, the participants took a mean time of 23 minutes ($SD = 8.3$ m) for creating their own mapping. Individual durations are listed in Table 10.12. The fastest patching was performed by participant 08, in only 06 minutes. The longest time for the mapping stage was used by participants 05 and 17 with 35 minutes.

The state of the mapping was continuously tracked throughout the mapping process. For every rendering parameter, the connected control parameter and the inversion factor were recorded. It is thus possible to fully reconstruct the process for a single participant and extract relevant statistical properties.

Table 10.12: Time used for the mapping process and total number of mapping changes for each participant.

Participant	05	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22
Minutes	35	20	6	12	12	27	30	32	14	22	25	35	18	28	27	20	29
Changes	126	31	9	57	19	36	82	45	31	35	74	100	23	43	25	55	73

The total number of changes made in the mapping, excluding inversion, is listed in Table 10.12 for each participant. This *mapping activity* counts the setting of new connections and the deletion of existing connections. The minimum necessary number of changes for meeting the requirements of the final mapping is 9, since every rendering parameter has to be connected at least once.

Overall, participants made a mean of 50.82 changes ($SD = 30.37$). Participant 08 made the fewest adjustments with the possible minimum of 9 changes. Most changes were made by participant 05, with a total of 126. The Pearson's correlation coefficient between the number of changes and the mapping duration is $r = 0.7$ ($p = 0.002$).

10.4.6 Individual Connections to Rendering Parameters

Figure 10.18 shows the matrix with the number of individual control parameters connected to each rendering parameter by all participants during the complete mapping stage. The higher the values, the more mapping possibilities of a rendering parameter were explored. The minimum value of 1 means that the parameter has been mapped to a control parameter once and has not been changed afterwards. The maximum value 10 means that a person has tried the combination of a rendering parameter with all control parameters. No empty values occur, since the guidelines required that all rendering parameters must be patched in the final mapping. Figure 10.12 shows a boxplot with the number of individual control parameters connected to the rendering parameters, grouped by participant.

Rendering Parameters	VP05	VP07	VP08	VP09	VP10	VP11	VP12	VP13	VP14	VP15	VP16	VP17	VP18	VP19	VP20	VP21	VP22
Spread	2	2	1	1	2	3	1	2	1	2	3	1	2	1	2	5	1
Distance	2	1	1	2	1	2	2	2	2	1	2	2	1	1	2	5	4
Elevation	2	1	1	1	1	1	3	1	2	1	3	4	1	1	2	1	2
Azimuth	3	1	1	1	1	1	2	1	2	1	2	2	1	1	1	1	2
Noise Level	5	2	1	2	2	1	4	4	3	4	3	5	2	3	2	3	4
Tonal Level	5	1	1	1	1	2	3	1	4	2	4	2	1	4	2	3	2
Intensity	4	1	1	1	1	3	1	2	1	2	3	2	2	3	2	1	3
Octave Offset	6	3	1	3	3	3	5	3	2	3	1	3	3	3	2	2	4
Pitch Offset	7	2	1	2	2	1	2	3	3	1	3	2	2	2	1	2	3

Figure 10.11: Matrix with the number of individual connections made to the rendering parameters for each participant.

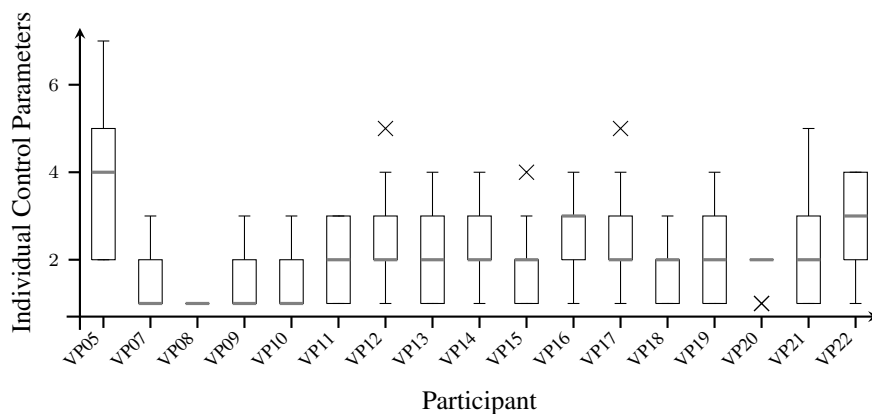


Figure 10.12: Number of individual control parameters connected to the rendering parameters, grouped by participant.

The number of individual connections is normalized for each participant by the absolute num-

ber of connections made by the participant. In order to allow an evaluation of the rendering parameters, they are grouped into three categories, namely *Pitch*, *Level* and *Spatial*. The group assignment of the individual parameters is shown in Table 10.13.

Table 10.13: Grouping of rendering parameters into three categories for the analysis of the exploration activity.

Pitch	Level	Spatial
Pitch Offset	Intensity	Azimuth
Octave Offset	Tonal Level	Elevation
	Noise Level	Distance
		Spread

The mean value of individual connections made by all participants is calculated for each category. Figure 10.13 shows a box plot with the number of individual connections from all participants, grouped by category.

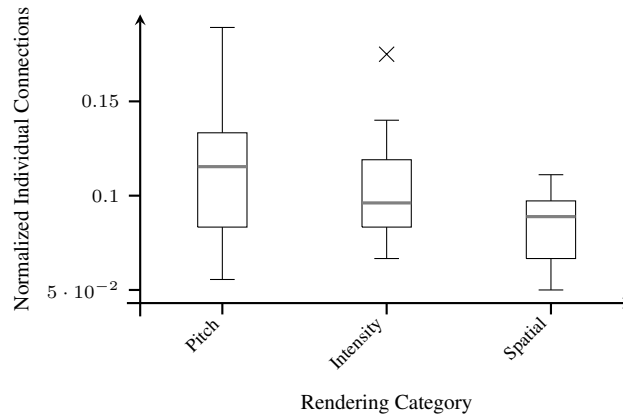


Figure 10.13: Normalized frequency of individual connections for the rendering categories.

Rank-Based ANOVA

Non-parametric statistical procedures were chosen due to the nature of the data and the small sample size. A 3x2 mixed design rank-based ANOVA (Wilcox, 2011) is performed for the within factor *Category* (Pitch, Level, Spatial), the between factor *Satisfaction* (High, Low) and the dependent variable *normalized individual connections* (NC). The *R* function `bwrnk()` from the related *R* package (Wilcox, 2020) is used for the calculation. Results show no significant effect of *Satisfaction* on the normalized individual connections and interactions.

10.4.7 Individual Connections to Control Parameters

Figure 10.14 shows the matrix with the number of individual rendering parameters connected to each control parameter by every participant throughout the complete mapping stage. The higher the entries, the more mappings of a parameter were explored. A 0 occurs if a participant did not connect a control parameter during the whole mapping stage. A value of 1 means that the control parameter has been mapped to a rendering parameter once and has not been changed afterwards. The maximum value of 9 means that a person has tried the combination of a control parameter with all rendering parameters.

Control Parameters	VP05	VP07	VP08	VP09	VP10	VP11	VP12	VP13	VP14	VP15	VP16	VP17	VP18	VP19	VP20	VP21	VP22
IMU: Roll	3	1	0	2	2	2	3	2	2	1	3	1	1	3	1	2	3
IMU: Pitch	4	1	1	2	1	1	3	1	2	1	2	3	1	2	2	1	4
IMU: Yaw	4	1	1	1	1	2	4	1	2	1	2	2	1	1	1	1	2
Ribbon: Swipe	3	2	1	2	1	0	1	1	1	1	1	2	1	1	1	3	3
Ribbon: Force	3	1	1	1	1	0	1	1	2	3	2	2	1	2	1	3	1
Ribbon: Pos	2	3	1	1	1	3	2	3	2	2	2	5	2	2	2	2	1
Pad: Pos	2	2	1	2	2	2	3	2	2	1	3	2	2	1	2	5	4
Pad: Force	5	1	1	1	2	4	4	2	2	2	4	2	4	2	4	5	3
Valve: Force	5	1	1	1	1	1	2	4	3	3	3	3	1	1	2	1	4
Valve: Binary	5	1	1	1	2	2	0	2	2	2	2	1	1	4	0	0	0

Figure 10.14: Matrix with the number of individual connections made to the control parameters by each participant.

Four participants (12, 20, 21, 22) did not make any use of the parameter *Valve: Binary* throughout the mapping stage. Subject 11 did not consider the two ribbon parameters force and position. Subject 08 did not use the parameter *IMU: Roll*. Figure 10.15 shows the number of individual connections to the control parameters, grouped by participant.

The number of individual connections is normalized for each participant by the absolute number of connections made by the participant. Afterwards, control parameters are grouped by control units, as listed in Table 10.14 and mean ratings are calculated for each participant in all groups. Figure 10.16 shows the normalized frequency of individual connections for the control units.

Table 10.14: Grouping of control parameters by unit for the analysis of the exploration activity.

Valves	Pad	Ribbon	IMU
Binary	Force	Force	Pitch
Force	Position	Position	Roll
		Swipe	Yaw

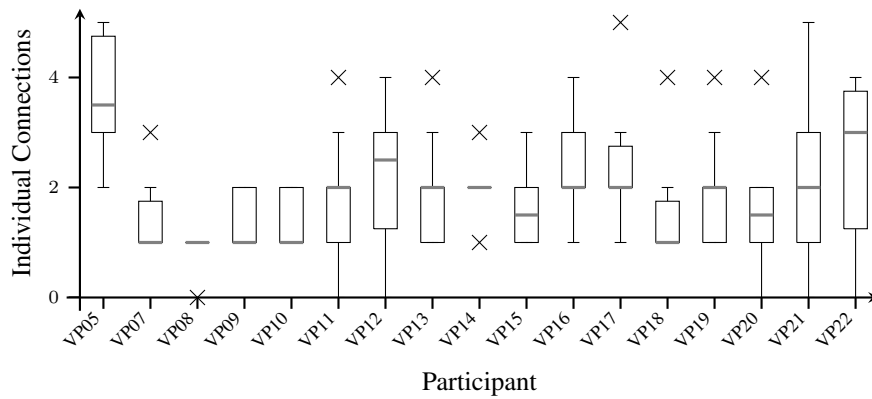


Figure 10.15: Number of individual rendering parameters connected to the control parameters, grouped by participant.

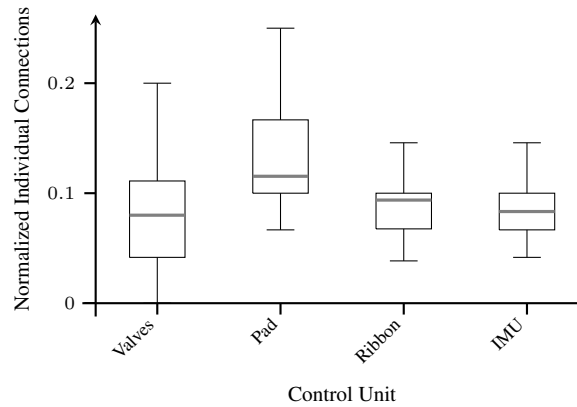


Figure 10.16: Normalized frequency of individual connections for the control units.

Rank-Based ANOVA

A 3x2 mixed design rank-based ANOVA (Wilcox, 2011) is performed for the within factor *Unit* (Valves, Pad, Ribbon, IMU), the between factor *Satisfaction* (High, Low) and the dependent variable *normalized individual connections* (NC). The *R* function `bwrnk()` from the related *R* package (Wilcox, 2020) is used for the calculation.

Results show no significant influence of the factor *Satisfaction* on the *normalized individual connections*. The factor *Unit* has a significant influence on the *normalized individual connections* ($F(3, 45) = 3.35$, $p = 0.022$). No significant interactions were found. A post-hoc Wilcoxon signed-rank test with false discovery rate (FDR) correction showed that significantly more individual connections were made for the *Pad* than for the *Valves* ($p = 0.049$), the *Ribbon* ($p = 0.036$) and the *IMU* ($p = 0.036$).

10.4.8 Final Mappings

The overall number of connections made by all participants for each control parameter in the final mappings is shown in Figure 10.17. With a total of 5 connections made, *Valve: Binary* is the least used control parameter. *Ribbon: Position* and *IMU: Pitch* have been connected 19 times, *Pad: Force* and *IMU: Yaw* 18 times. A χ^2 test shows no significant divergence from equal distribution ($p = 0.334$).

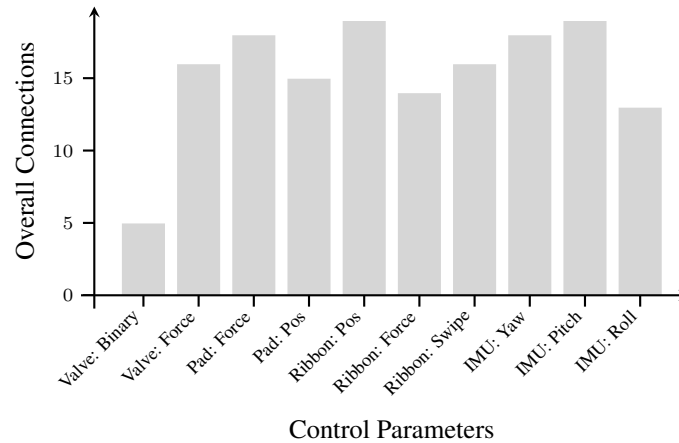


Figure 10.17: Mapping frequency for the control parameters over all participants in the final mappings.

Figure 10.18 shows the matrix with the absolute frequencies of connections made between control and rendering parameters by all participants in their final mappings. Due to the mapping guidelines, which allow one-to-many connections and exclude many to one connections, the sum of each column is 17, since every participant had to map all rendering parameters only once. The sums of the columns are individual, depending on the above presented use frequency of the relevant control parameter.

χ^2 values are calculated for the distribution of all control parameters to the different rendering parameters. The results are shown in Table 10.15. With exception of *Valve: Binary*, *Ribbon: Position* and *IMU: Roll*, all control parameters significantly diverge from uniform distribution.

	Valves		Pad		Ribbon			IMU		
	Binary	Force	Force	Pos	Pos	Force	Swipe	Yaw	Pitch	Roll
p	0.191	0.002	0.021	0.032	0.531	0.020	< 0.001	< 0.001	< 0.001	0.189

Table 10.15: χ^2 tests for all control parameters

Most prominent are the mappings between the control parameter *IMU: Yaw* and the spatialization parameter *Azimuth* (13/18) as well as between *IMU: Pitch* and *Elevation* (12/19). The control parameter *Ribbon: Swipe* is connected to the synthesis parameter *Octave Offset* 9 out of 16 times. The control parameter *Valve: Force*, was most frequently mapped to the *Intensity* (6/16), followed by the *Tonal Level* (5/16). *Pad: Position* was most frequently mapped to the *Pitch Offset* (6/15). *Ribbon: Pos* was often used, yet not preferred for specific rendering parameters.

Based on the individual mapping matrices M_i of the participants, a similarity matrix S is calcu-

Control Parameters	IMU: Roll	1	2	0	1	0	3	0	4	2
	IMU: Pitch	0	0	0	0	0	0	12	6	1
	IMU: Yaw	0	1	1	1	1	13	1	0	0
	Ribbon: Swipe	1	9	1	1	2	0	0	1	1
	Ribbon: Force	1	1	1	3	6	0	0	1	1
	Ribbon: Pos	3	2	4	1	1	0	2	2	4
	Pad: Pos	6	1	0	2	2	0	2	2	0
	Pad: Force	2	0	4	1	5	0	0	1	5
	Valve: Force	1	0	6	5	0	1	0	0	3
	Valve: Binary	2	1	0	2	0	0	0	0	0
	Pitch Offset									
	Octave Offset									
Rendering Parameters										
	Intensity									
	Tonal Level									
	Noise Level									
	Azimuth									
	Elevation									
	Distance									
	Spread									

Figure 10.18: Mapping matrix, showing the total number of connections over all participants.

lated between all participants. It is defined by the sum of the Hadamard product, with the number of participants N_p :

$$S_{i,j} = \sum_{i=1}^{N_p} \sum_{j=1}^{N_p} M_i \circ M_j \quad (10.1)$$

Since each individual mapping matrix contains exactly 9 elements with the value 1, the similarity value is 9 for identical mappings and thus across the diagonal. A value of 0 means that two participants did not use a single identical connection in their mapping. The maximum similarity is found between participants 05 and 22, with a value of 7. A similarity of 6 is found between participants 07 and 20, 15 and 16 as well as between 08 and 21. 8 pairings result in a similarity of 5. A single similarity score is calculated for each participant as the mean similarity to all other participants. Results are presented in Table 10.16. Highest values are above a similarity value of 3. Participant 19 shows the lowest score with 0.19.

Table 10.16: Mean similarity score for each participant.

Participant	05	07	08	09	10	11	12	13	
Similarity	3.12	2.75	3.06	2.56	2.75	2.62	1.19	2.12	
Participant	14	15	16	17	18	19	20	21	22
Similarity	0.94	2.56	2.00	2.75	2.44	0.19	3.12	2.56	3.12

Participant	VP05	VP07	VP08	VP09	VP10	VP11	VP12	VP13	VP14	VP15	VP16	VP17	VP18	VP19	VP20	VP21	VP22
VP22	7	3	3	5	4	4	1	4	1	3	3	4	2	0	3	3	9
VP21	3	2	6	2	4	3	2	2	0	3	2	2	4	0	3	9	3
VP20	4	6	4	2	4	3	1	3	1	4	3	5	4	0	9	3	3
VP19	0	0	0	1	0	0	0	1	0	0	0	1	0	9	0	0	0
VP18	3	2	4	2	4	3	2	4	1	2	1	1	9	0	4	4	2
VP17	3	5	3	4	2	2	0	1	2	5	4	9	1	1	5	2	4
VP16	2	2	3	2	1	0	1	0	2	6	9	4	1	0	3	2	3
VP15	2	3	5	2	2	1	0	1	2	9	6	5	2	0	4	3	3
VP14	1	1	1	2	0	0	1	0	9	2	2	2	1	0	1	0	1
VP13	5	2	2	2	3	3	1	9	0	1	0	1	4	1	3	2	4
VP12	1	1	2	1	2	3	9	1	1	0	1	0	2	0	1	2	1
VP11	4	4	4	3	5	9	3	3	0	1	0	2	3	0	3	3	4
VP10	4	3	4	2	9	5	2	3	0	2	1	2	4	0	4	4	4
VP09	5	4	2	9	2	3	1	2	2	2	2	4	2	1	2	2	5
VP08	3	3	9	2	4	4	2	2	1	5	3	3	4	0	4	6	3
VP07	3	9	3	4	3	4	1	2	1	3	2	5	2	0	6	2	3
VP05	9	3	3	5	4	4	1	5	1	2	2	3	3	0	4	3	7

Figure 10.19: Matrix with mapping similarities between participants.

10.5 Discussion

10.5.1 Sample and Surveys

Since participants were recruited through the lists of the Audio Communication Group, a homogeneous expert group was expected. Based on the self-assessment of experience in spatial audio and with electronic musical instruments, the sample can be considered a group with above average expertise. Four participants agreed completely that they have experience with both spatial audio and electronic musical instruments. Only participant 19 scored low for both expertise items.

Results of the GMSI self-report questionnaire can be compared to the large sample from the GMSI data set. In general, the mean ratings of the sample in this study are similar to these benchmarks. Although performing slightly better in some categories, the high standard deviations do not allow a ranking. Since the 147,633 participants from the reference study represent a cross section, the sample of this study can be considered average in all scales of musical sophistication.

The UEQ results for the synthesis system can be compared to the benchmarks included in the material of the questionnaire. It performs well in the hedonic qualities and shows weaknesses in the pragmatic qualities. Although the system should be as easy to use as possible, low ratings in perspicuity and efficiency were expected, since it is complex and can not be fully grasped at the first moment. Stimulation and novelty are rated *good* and *excellent*, meaning that the system is appreciated in non-task related aspects. It is likely that these ratings are common for many musical instruments and especially for digital music instruments. They are not necessarily easy to learn and a considerable amount of practice is needed before they are mastered and thus efficient and perspicuous. Instruments with means for expressive control are more prone to faulty operation since even small changes in control values affect the sound. Since the data for the benchmark was gathered from business applications and related products, it results in higher values for the pragmatic qualities.

10.5.2 Text Response Examples

Responses to the open questions were scanned for frequent remarks. Three participants noted that the granted time was too short. Indeed, the limit of 30 minutes is rather restrictive and could cause unnecessary pressure, especially for users with less experience.

Since the interface is the most graspable component of the system, most remarks referred to specific sensors. Four participants were not satisfied with the sensitivity and range of the force-sensing units. They were considered not sensitive enough by three participants. The lack of a possibility for scaling the sensor data and defining minimum and maximum values was expressed by two participants. Indeed these are known weaknesses of the system as used in the user study.

Three participants mentioned the pad as a pleasant aspect of the system, whereas one participant pointed it out as unpleasant. Two people noticed problems of opposing valves and ribbon, since they can not be operated, independently.

Participants had less comments referring to the synthesis system. According to five responses, the ability to control the spacial aspects of sound was the most appealing aspect of the system. However, two participants wished for a clear decoupling between spatial and timbral control. Three participants suggested the use of different sounds for spatialization. One pointed out that sounds with more transients would be better for sound localization.

10.5.3 Final Mappings

The final mappings from the participants reveal that there are no significant preferences of specific control parameters. The rare use of the control parameter *Valve: Binary* was expected, since it was already hard-patched in the system. Clear preferences can be found for specific connections in the mapping matrix. Most prominent is the frequent connection between IMU parameters and rendering parameters from the spatial category. This coupling confirms the design, since *IMU: Yaw* was used to control the *Azimuth* and *IMU: Pitch* was used for the *Elevation* during development and testing. However, the frequent use of IMU parameters for spatial control could also be biased by the arrangement of the parameters in the mapping interface. Since *Ribbon:Swipe* was designed to control the *Octave Offset*, the high mapping frequency between these parameters is obvious.

The control parameter *Valve: Force* is not used above average for modulating the pitch, as it was intended since the very first design studies of the musical interface. Already during the development it became obvious that this connection does not result in a practical control. When changing the fingering, the force is changed, leading to undesired pitch slides. But the force parameter of the valves was predominantly used for controlling the intensity and the tonal level. Overall, there is a tendency to map force-based control parameters to rendering parameters from the *Level* category. This seems plausible, since excitation is usually associated with the exertion of force.

According to the matrix of mapping similarities in Figure 10.19, several pairings of participants show a high similarity in the final mappings. This indicates that there are sets of preferred mappings. Participant 19 differs notably from the rest of the sample, with a similarity score of 0.19. This participant also has the lowest mean experience with electronic musical instruments and spatial sound and could be considered an outlier.

10.5.4 Mapping Process

Results of the mapping process provide insight into the patching activity for individual participants and system parameters, respectively parameter groups and allow conclusions on the mapping strategies. Based on the correlation analysis, participants with more changes in the connections take longer for finishing the mapping stage. The following paragraphs discuss the results with regard to the system parameters.

Rendering Parameters

The rank-based ANOVA did not show a significant influence of the category of synthesis parameters on the number of individual connections made to the parameters. The statistical evaluation did not reveal a significant relation between the satisfaction of the participants and the number of individual connections made to the parameter categories. According to these results, all parameter categories were considered by the participants in the same way. This can be explained by the fact that all rendering parameters had to be used in the final mappings.

Control Parameters

According to the rank-based analysis of variance, parameters from the unit *Pad* have been connected to more individual rendering parameters during the mapping process than those from the *Valves*, the *Ribbon* or the *IMU*. Since multiple remarks in the survey mentioned the Pad as a pleasant aspect, it can be hypothesized that this preference results in an extensive exploration of its parameters. Four participants did not make any use of the control parameter *Valve: Binary*. It is plausible that this parameter was explored less than others, since it is hard-patched to the pitch selection.

As for the rendering parameter categories, no significant relation was revealed between the satisfaction of the users and the number of individual parameters they connected to the units. This indicates that users are able to reach a satisfying mapping without exploring a large variety of possibilities.

10.5.5 Mapping Strategies

Based on the continuous tracking of connections, the mapping procedure can be visualized for individual participants. Errors in operation can cause unintended disconnections in the recordings. Such instances are clearly detectable, since all mapping connections are removed and restored at the same time. Gaps were removed manually for participant 08 and 15. Three examples are displayed and discussed in the following, focusing on different mapping strategies.

Participant 08

Participant 08 is a *high confidence mapper* with high expertise in both spatial audio and electronic musical instruments. Throughout the mapping stage, the connections were changed only 09 times and the task was completed in 6 minutes. All parameters have been set once and were not changed, afterwards.

The associated mapping process, visualized in Figure 10.20, reveals a structured approach. This can also be backed by the explanation of the mapping process given by the participant in the survey. Mapping was organized by control parameter category, starting with the *Level* parameters of the rendering system, followed by *Pitch* parameters and the *Spatial* parameters.

The final mapping of participant 08 has the highest similarity score of 3.12, indicating a conventional mapping. According to the ratings from the twelve attributes from the survey, the

participant belongs to the group with a high satisfaction, despite being quick and exploring little.

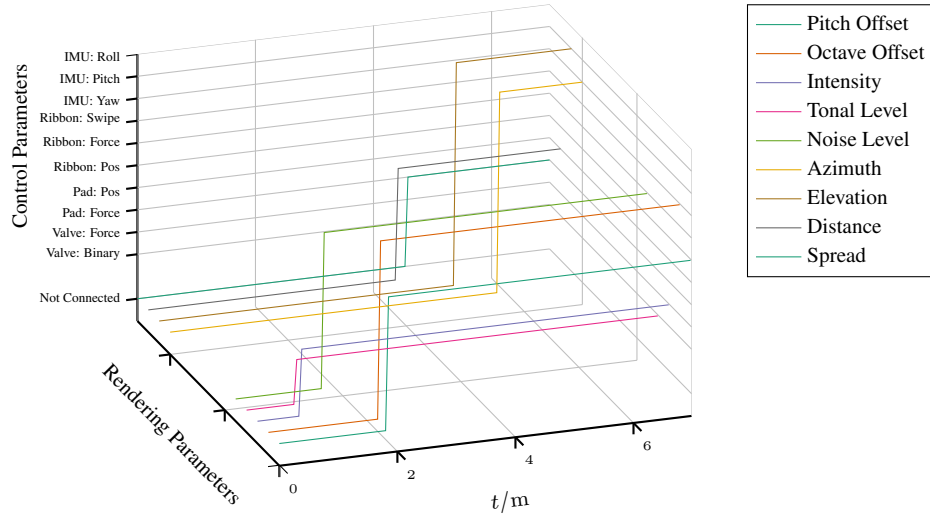


Figure 10.20: Mapping process of participant 08.

Participant 14

With 31 connection changes in 14 minutes, participant 14 can be considered moderate in the mapping activity, taking less time than the average of the sample. The number of individual connections made for the rendering and control parameters is average.

Figure 10.21 shows the mapping process of participant 14. The mapping trajectories indicate that different connections have been explored for most parameters. Most of the established connections were left for several minutes, allowing the evaluation of the resulting effects. This is also reflected in the descriptions of the mapping procedure from the survey. This participant is described as an *exploring mapper*.

The final mapping result of participant 14 is rather unique, with a similarity score of 0.94. Although IMU parameters were used for the spatial rendering parameters, they were assigned differently. According to the 12 ratings on the rendering attributes, this participant belongs to the group with a high satisfaction.

Participant 05

Participant 05 made 126 changes during the mapping stage, which is the maximum number within this study. According to the individual connections made to rendering and control parameters, participant 05 explored more possibilities than any other participant. The mapping process was finished after 35 minutes, after a second notification through the test management.

The corresponding mapping process is plotted in Figure 10.22. Connection trajectories of single control parameters show several connections which are removed shortly after being established.

According to the descriptions of the mapping process from the survey, participant 05 followed an 'unconscious' approach. In the open response part of the survey it was also noted that the

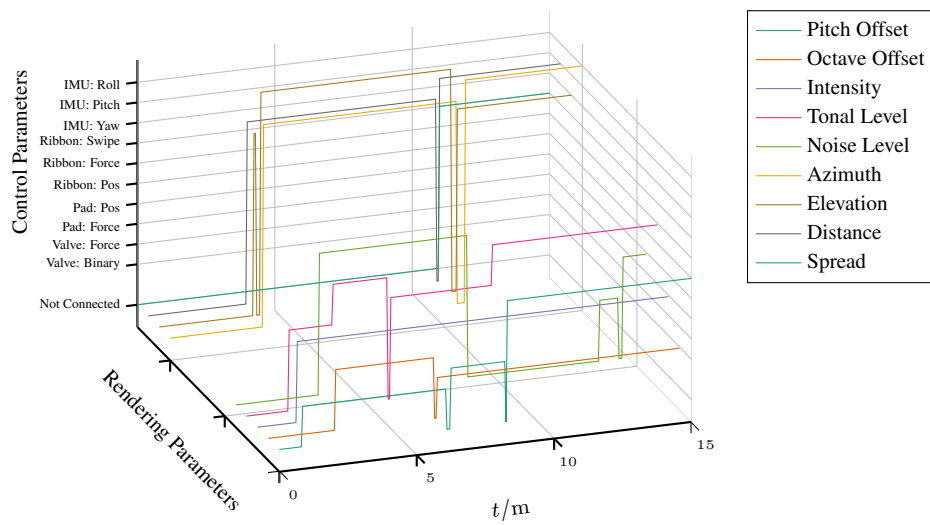


Figure 10.21: Mapping process of participant 14.

system could be 'simpler and easier to understand'. Considering these remarks, this participant is regarded a *confused mapper*, obviously overstrained with the mapping task and the system. According to the 12 attributes, participant 05 belongs to the group with a low satisfaction value.

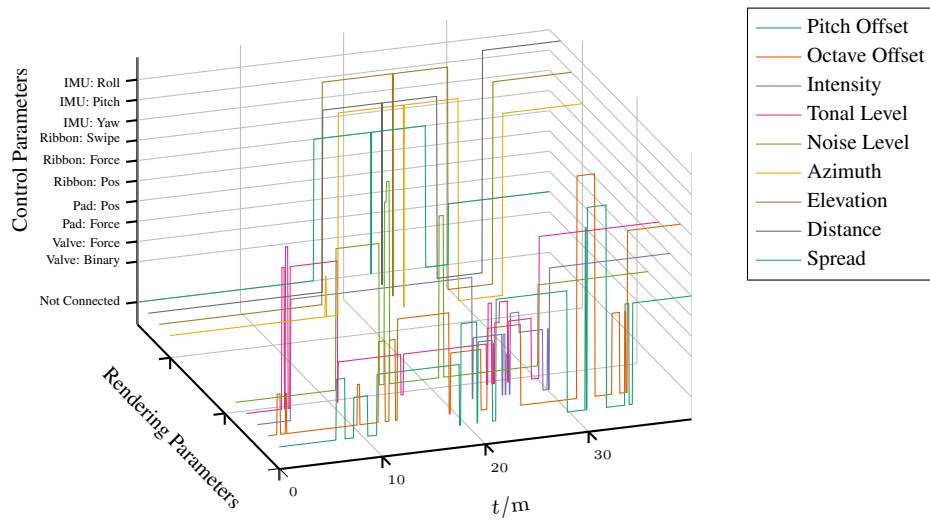


Figure 10.22: Mapping process of participant 05.

Conclusion

Starting with a focus on signal processing, the project presented in this thesis developed into a full system for spatial sound synthesis. While its use in actual musical performances was of course always considered, it is first and foremost a research tool and object. Especially the combination of spectral modeling, spatial sound synthesis and expressive control can be considered a distinguishing feature. By swapping single components, it can be used to investigate more general questions in the field of music and human-computer interaction.

11.1 The Modeling Approach

Although the synthesis approach is based on the analysis of instrument sounds, it was not intended to result in a highly plausible emulation of a violin. The model should rather learn specific aspects from the original instrument, with an emphasis on spectral envelope modulations and stochastic timbre fluctuations. The synthesis of these aspects is considered a key feature to vivid and expressive synthesis results. Descriptive and numeric evaluation of synthesis results could confirm that the *statistical spectral modeling* system reproduces spectral envelope modulations, and intensity-based timbre changes.

After delivering a proof of concept for the violin, it would be of interest to model additional instruments. An extension for instruments without continuous excitation could be possible and would complete the approach. Pitch and intensity are the basic parameters of melody instruments and allow actual expressive play. Still, additional dimensions for the timbre of the synthetic sounds would considerably increase the expressive capabilities. The timbre plane and the interpolated Markovian inverse transform sampling can be extended by additional control parameters, resulting in a system capable of timbre manipulation and morphing.

As already realized in related projects, the modeling stage could be fed with complete interpretations instead of prepared sounds. This would render the design of custom sample libraries

unnecessary and new instrument models could be generated, rapidly. This possibility was considered but dropped early in the project, since it would have caused a shift from the instrument design to music information retrieval. Now that a proof of concept was delivered for the modeling approach, this step would be very interesting.

Continuing this line of thought, it would be even more interesting to create models by design rather than by analysis. The next step in the development of the modeling algorithm is thus the replacement of the fixed distributions with parametric ones. This would allow the dynamic change of statistical properties through metaparameters, resulting in novel sounds.

11.2 Synthesis Software

The recent version of the synthesis software implements the *MITSSy* algorithm with the necessary extensions for spatial sound synthesis. It performs well for the intended purposes but still needs improvement. Since it can be compiled and used on most standard Linux systems, the user group can be expanded, allowing further development through public repositories. Porting the software to other operating systems should be possible, especially for Mac systems.

At this point the synthesis frame size is still coupled to the buffer size of the audio interface. A decoupling will result in constant synthesis quality for different sound card settings. A major drawback of the recent version is the slow start up, caused by the use of plain text files and `yaml-cpp` for reading the model data. During the major design phase these data formats were kept for allowing quick debugging and evaluation. Switching to a more efficient format for the model is absolutely necessary now, for encouraging the use of the software.

In the recent implementation the synthesizer handles the 80 partials of the violin model without issues. When considering instruments with lower fundamental frequencies, the number of partials needs to be increased. To this point, the algorithm has not been pushed further, but it is unlikely to cope with several hundred partials. For the sake of performance, the frequency domain synthesis approaches should be reconsidered. As discussed in the implementation details, this would complicate the individual use of single partials. This problem could be solved by using a direct connection between frequency bins and spherical harmonics, as implemented in examples from the related work. In the long run this would require the embedding of a spatial rendering algorithm in the synth, which would also improve the performance.

Besides the implementation of the *MITSSy* algorithm, the *GLOOO* synthesis software constitutes a useful framework for additional synthesis techniques. Voice management, control structures and the multichannel capabilities are well suited for spatial granular synthesis and other approaches. Through the modular design of the software, additional algorithms could be inherited from the `SingleVoice` class.

11.3 Spatial Synthesis

The interactive spectro-spatial synthesis is made possible by combining the synthesis software with existing software for spatial sound rendering. This modularity can be considered a benefit when working on professional Linux audio systems, which rely on JACK. But using external rendering tools was also identified as a performance bottleneck. The recent combination with the SuperCollider Ambisonics rendering system showed significant improvements but was not tested exhaustively. Including a rendering stage in the synthesis software would be most efficient and offer additional means for spectro-spatial sound synthesis. However, this would be another major project.

So far, the spatial aspect of the system is rooted in a point source paradigm. The advantage of this approach is that it is agnostic of the reproduction system, since an arrangement of point sources can be rendered to any loudspeaker configuration or headphones. Furthermore, it allows the creation of a virtual sound source with spatial extent and a heterogeneous spectral distribution. This concept is comprehensible for most users and offers immediate access. Nevertheless, the exploration of less conservative approaches to spatialization should be considered in the future.

The GLOOO synthesis software offers different modes of spectral dispersion for different spectro-spatial distributions. Ongoing work focuses on a perceptual evaluation of these modes, in combination with different geometries of the virtual sound source. The quasi spherical model implemented for the user study can only be considered a first draft. A structured exploration of basic geometries in combination with different spectral distributions could provide insight into the actual added value of spectro-spatial synthesis.

11.4 Control & Mapping

Throughout the project, the haptic interface has been improved and developed. The revised second version shows promising features controlling spectral sound synthesis and other electronic musical instruments. Although developed for the use with the GLOOO synthesis system, it can be regarded an independent, versatile musical interface. Especially the interdisciplinary development and the profound knowledge from classical instrument manufacturing produced valuable components, such as the valves or the excitation pad. Regardless of the complete device, they are considered highly functional and widely applicable. However, their combination in the BIN-BONG can still be improved in several aspects.

The ribbon controller was an emergency solution, since the initially installed capacitive sensors did not work reliably in the final stages of the MKII development. It provides additional possibilities, which could be beneficial in certain applications but showed to be inadequate for the octave selection. Mounted opposed to the force-sensitive valves, the two sensor units are interacting in an undesired way. The original piezo switches from the first version were replaced, although they worked perfectly as octave switches. The original reason for this change was the high price of the piezo switches.

The graphical mapping environment in Pure Data completes the system, allowing the customization of the control for specific scenarios. Originally conceived for the mapping experiment, it showed to be a powerful solution beyond the user study and is included in the software package. The BINBONG has since been applied in a different setup for live electronics, due to the adaptability. A newer version of the mapping software uses pre-defined abstractions for inversion, scaling and offsets. Only the integration of many-to-one mappings has not been tackled, yet.

Although Pure Data is a powerful mapping environment for experienced users, is rather complicated for novice users. Some participants in the user study had problems patching, which could be observed from the control room. Even with a decent mouse it requires a certain precision for hitting the rather small inlets and outlets of the Pure Data elements. Since connections had to be changed in the edit mode, users could accidentally drag the sub-patches for the control units. From this point of view it would be practical to implement a simple graphical mapping environment for this system or to test existing mapping tools from the NIME community.

11.5 User-Defined Mappings

The complete instrument was applied in a user study, focusing on user-defined mappings for spectro-spatial sound synthesis. This allowed the evaluation of the system as a whole, and provided insights on individual components. Even though the results from the study showed the need for several improvements, the system was appreciated by the majority.

While complex, the mapping task was mastered by most participants and also considered rather appealing. This can be explained by the strong expertise of the sample. One person with notably less experience was overstrained with the task. Final mappings indicated preferences for connections, which confirm certain design decisions. More importantly, this procedure allowed the exploration of mapping strategies of the users. Since the process was fully captured, the individual mapping strategies of participants could be analyzed and compared. Using the gathered data from the system and the surveys, it was possible to identify individual characteristics. Participants showed differences in the patching activity. Some finished their mapping in six minutes, whereas others took more than half an hour. The statistical analyses did not reveal any influence of the patching activity on the satisfaction of the participants with their mapping. Different strategies result in satisfying configurations.

One aspect left for future evaluations is the recorded data of the sensors, respectively the control parameters, during the mapping stage and the gesture part. In combination with the analyses presented in this thesis it could help to understand how the use of the sensors interacts with the mapping strategies.

Ultimately, the complete system in combination with the user-centered study is considered a valuable tool for further experiments and upcoming research questions. Framework and procedure are well suited for examining other interfaces, synthesis approaches and spatialization techniques in a real-time scenario.

References

- Abe, M. & Smith, J. (2005). AM/FM rate estimation for time-varying sinusoidal modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (Vol. 3, pp. 201–204). Philadelphia, Pennsylvania, USA.
- Alonso Moral, J. & Jansson, E. (1982). Input admittance, eigenmodes and quality of violins. In *Speech, Music and Hearing Quarterly Progress and Status Report* (Vol. 23, 2-3, pp. 60–75). KTH Royal Institute of Technology.
- Alpaydin, E. (2010). *Introduction to machine learning*. MIT Press.
- Andersen, T. H. & Jensen, K. (2004). Importance and representation of phase in the sinusoidal model. *Journal of the Audio Engineering Society*, 52(11), 1157–1169.
- Anson, E. (1982). The device model of interaction. In *Proceedings of the 9th Annual Conference on Computer Graphics and Interactive Techniques* (pp. 107–114). New York, USA.
- Ardaillon, L., Degottex, G., & Roebel, A. (2015). A multi-layer F0 model for singing voice synthesis using a B-spline representation with intuitive controls. In *Annual Conference of the International Speech Communication Association (Interspeech)*. Dresden, Germany.
- Arfib, D., Couturier, J. M., & Kessous, L. (2005). Expressiveness and digital musical instrument design. *Journal of New Music Research*, 34(1), 125–136.
- Arroabarren, I., Zivanovic, M., Bretos, J., Ezcurra, A., & Carlosena, A. (2002). Measurement of vibrato in lyric singers. *IEEE Transactions on Instrumentation Measurement*, 51(4), 1529–1534.
- Arroabarren, I., Zivanovic, M., & Carlosena, A. (2002). Analysis and synthesis of vibrato in lyric singers. In *Proceedings of the 11th European Signal Processing Conference* (Vol. 107, pp. 397–400). Toulouse, France.

REFERENCES

- Arroabarren, I., Zivanovic, M., Rodet, X., & Carlosena, A. (2003). Instantaneous frequency and amplitude of vibrato in singing voice. In *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 537–540). Hong Kong, China.
- Austin, L. (2000). Sound diffusion in composition and performance: An interview with Denis Smalley. *Computer Music Journal*, 24(2), 10–21.
- Baalman, M. (2010). Spatial composition techniques and sound spatialisation technologies. *Organised Sound*, 15, 209–218.
- Baalman, M. A. & Plewe, D. (2004). WONDER - a software interface for the application of wave field synthesis in electronic music and interactive sound installations. In *Proceedings of the International Computer Music Conference (ICMC)*. Miami, Florida.
- Baalman, M., Bovermann, T., de Campo, A., & Negrao, M. (2014). Modality. In *Joint proceedings of the 40th International Computer Music Conference, and the 11th Sound and Music Computing Conference*. Athens, Greece.
- Baecker, R. (1980). Human-computer interactive systems: A state-of-the-art review. In *Processing of Visible Language* (pp. 423–443). Springer.
- Baisnée, P.-F., Barriere, J.-B., Koechlin, O., & Rowe, R. (1986). Real-time interaction between musicians and computer: Live performance utilisations of the 4x musical workstation. In *Proceedings of the International Computer Music Conference (ICMC)*. Den Haag, The Netherlands.
- Bartkowiak, M. & Zernicki, T. (2011). A non-time-progressive partial tracking algorithm for sinusoidal modeling. In *Proceedings of the 131st Audio Engineering Society Convention*. Audio Engineering Society. New York, USA.
- Bathey, B. (2004). Bézier spline modeling of pitch-continuous melodic expression and ornamentation. *Computer Music Journal*, 28(4), 25–39.
- Battier, M. (2015). Recent discoveries in the spatial thought of early musique concrète. In M. Brech & R. Paland (Eds.), *Compositions for audible space. The early electroacoustic music and its contexts. Music and sound culture* (pp. 123–36).
- Beauchamp, J. W. (1993). Unix workstation software for analysis, graphics, modification, and synthesis of musical sound. In *Proceedings of the Audio Engineering Society Convention*. Audio Engineering Society. Berlin, Germany.
- Beauregard, G. T. (1991). *Rethinking the design of wind controllers* (Doctoral dissertation, Dartmouth College).

REFERENCES

- Ben-Kiki, O., Evans, C., & Net, I. d. (2020, September 20). YAML ain't markup language (YAML™) version 1.2. Retrieved from <http://yaml.org>
- Bencina, R. (2005). The Metasurface: Applying natural neighbour interpolation to two-to-many mapping. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 101–104). Vancouver, Canada.
- Benson, C., Manaris, B. Z., Stoudenmier, S., & Ward, T. (2016). Soundmorpheus: A myoelectric-sensor based interface for sound spatialization and shaping. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 332–337). Brisbane, Australia.
- Berg, J. & Rumsey, F. (2003). Systematic evaluation of perceived spatial quality. In *Proceedings of the 24th AES International Conference on Multichannel Audio*. Banff, Canada.
- Bevilacqua, F., Müller, R., & Schnell, N. (2005). MnM: A MAX/Msp mapping toolbox. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 85–88). Vancouver, Canada.
- Böhlke, L. & Ziemer, T. (2017). Perceptual evaluation of violin radiation characteristics in a wave field synthesis system. In *Proceedings of Meetings on Acoustics* (Vol. 30, 1, p. 035001). ASA. New Orleans, USA.
- Bonada, J. (2004). High quality voice transformations based on modeling radiated voice pulses in frequency domain. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)* (Vol. 3, pp. 291–295). Naples, Italy.
- Bonada, J., Loscos, A., & Kenmochi, H. (2003). Sample-based singing voice synthesizer by spectral concatenation. In *Proceedings of Stockholm Music Acoustics Conference* (pp. 1–4). Stockholm, Sweden.
- Bonada, J., Serra, X., Amatriain, X., & Loscos, A. (2011). Spectral processing. In U. Zoelzer (Ed.), *DAFX: Digital audio effects* (2nd ed., pp. 393–445). John Wiley & Sons.
- Bonada, J. & Serra, X. (2007). Synthesis of the singing voice by performance sampling and spectral models. *IEEE Signal Processing Magazine*, 24(2), 67–79.
- Bongers, B. (2000). Interaction theory and interfacing techniques for real-time performance. In *Trends in Gestural Control of Music* (pp. 41–70). Ircam, Paris, France.
- Bowler, I., Purvis, A., Manning, P. D., & Bailey, N. (1990). On mapping n articulation onto m synthesiser-control parameters. In *Proceedings of the International Computer Music Conference (ICMC)*. Glasgow, Scotland.

REFERENCES

- Brandtsegg, Ø., Saue, S., & Johansen, T. (2011). A modulation matrix for complex parameter sets. In *Proceedings of the Conference on New Interfaces for Musical Expression (NIME)* (pp. 316–319). Oslo, Norway.
- Brech, M. & von Coler, H. (2014). The Halaphon and its use in Luigi Nono's 'Prometeo' in Venice. In *Proceedings of the 9th Conference on Interdisciplinary Musicology (CIM)*. Berlin, Germany.
- Brech, M. & von Coler, H. (2015). Aspects of space in Luigi Nono's Prometeo and the use of the Halaphon. In M. Brech & R. Paland (Eds.), *Compositions for audible space* (pp. 193–204). Music and Sound Culture. transcript.
- Bregman, A. S. (1990). *Auditory Scene Analysis: The perceptual organization of sound*. MIT Press.
- Bresin, R. (2001). Articulation rules for automatic music performance. In *Proceedings of the International Computer Music Conference (ICMC)*. Havana, Cuba.
- Bresson, J. (2012). Spatial structures programming for music. In *Proceedings of the Spatial Computing Workshop (SCW) – co-located w. Autonomous Agents and Multi Agent Systems (AAMAS)*. Valencia, Spain.
- Brilmayer, B. (2014). *Das Trautonium. Prozesse des Technologietransfers im Instrumentenbau* (Doctoral dissertation, Universität Augsburg).
- Brown, D., Nash, C., & Mitchell, T. (2017). A user experience review of music interaction evaluations. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Copenhagen, Denmark.
- Brown, D., Nash, C., & Mitchell, T. (2018). Understanding user-defined mapping design in mid-air musical performance. In *Proceedings of the 5th International Conference on Movement and Computing* (p. 27). Genoa, Italy.
- Burns, C., Serafin, S., & Burtner, M. (2003). Musical applications of generalised multichannel digital waveguides. In *Proceedings of the Stockholm Music Acoustics Conference*. Akademisk Forlag. Stockholm, Sweden.
- Burred, J., Röbel, A., & Rodet, X. (2006). An accurate timbre model for musical instruments and its application to classification. In *Proceedings of the 1st Workshop on Learning the Semantics of Audio Signals*. Athens, Greece.
- Buxton, W. (1983). Lexical and pragmatic considerations of input structures. *ACM SIGGRAPH Computer Graphics*, 17(1), 31–37.

REFERENCES

- Caetano, M., Kafentzis, G., Degottex, G., Mouchtaris, A., & Stylianou, Y. (2013). Evaluating how well filtered white noise models the residual from sinusoidal modeling of musical instrument sounds. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)* (pp. 1–4). New Paltz, NY, USA.
- Camacho, A. (2007). *Swipe: A sawtooth waveform inspired pitch estimator for speech and music* (Doctoral dissertation, University of Florida, Gainesville, FL, USA).
- Cann, S. & Rausch, K. P. (2007). *Sample this* (3rd ed.). Coombe Hill Pub.
- Cannam, C., Landone, C., & Sandler, M. (2010). Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files. In *Proceedings of the 18th ACM international conference on Multimedia* (pp. 1467–1468). ACM. New York, USA.
- Caramiaux, B., Bevilacqua, F., & Schnell, N. (2009). Towards a gesture-sound cross-modal analysis. In *Proceedings of the International Gesture Workshop* (pp. 158–170). Springer.
- Carlson, C., Marschner, E., & McCurry, H. (2011). The sound flinger: A haptic spatializer. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Oslo, Norway.
- Carpentier, T. (2016). Panoramix: 3d mixing and post-production workstation. In *Proceedings of the International Computer Music Conference (ICMC)*. Utrecht, Netherlands.
- Chadabe, J. (2002). The limitations of mapping as a structural descriptive in electronic instruments. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 1–5). Dublin, Ireland.
- Chamberlin, H. (1985). *Musical applications of microprocessors* (Second Edition). Hayden Books.
- Charpentier, F. & Stella, M. (1986). Diphone synthesis using an overlap-add technique for speech waveforms concatenation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (Vol. 11, pp. 2015–2018). Tokyo, Japan.
- Chowning, J. (2011). Turenas: The realization of a dream. In *Proceedings of the 17th Journées d'Informatique Musicale*. Saint-Etienne, France.
- Curtin, J. & Rossing, T. D. (2010). Violin. In T. Rossing (Ed.), *The Science of String Instruments* (pp. 209–244). Springer.
- D'Alessandro, N. & Dutoit, T. (2007). HandSketch Bi-Manual Controller - investigation on expressive control issues of an augmented tablet. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 78–81). New York, USA.

REFERENCES

- da Silveira, G. O. (2018). The XT Synth: A new controller for string players. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 43–44). Blacksburg, VA, USA.
- Dahlstedt, P. (2009). Dynamic mapping strategies for expressive synthesis performance and improvisation. In S. Ystad, R. Kronland-Martinet, & K. Jensen (Eds.), *Computer music modeling and retrieval. genesis of meaning in sound and music* (Vol. 5493, pp. 227–242). Lecture Notes in Computer Science. Springer Berlin Heidelberg
- Dannenbergh, R. B. & Derenyi, I. (1998). Combining instrument and performance models for high-quality music synthesis. *Journal of New Music Research*, 211–238.
- Das, O., Smith, J., & Chafe, C. (2017). Real-time pitch tracking in audio signals with the extended complex kalman filter. In *Proceedings of the 20th International Conference on Digital Audio Effects (DAFx)*. Edinburgh, UK.
- Davies, Paul. (2020). JACK API. Retrieved April 20, 2020, from <https://jackaudio.org/>
- Davies, H. (1996). A history of sampling. *Organised Sound*, 1(1), 3–11.
- Davis, T. & Karamanlis, O. (2007). Gestural control of sonic swarms: Composing with grouped sound objects. In *Sound and Music Computing*.
- Dawes, Beman and Abrahams, David. (2020). Boost C++ Library. Retrieved April 20, 2020, from <https://www.boost.org/>
- De Campo, A. (2014). Lose control, gain influence – concepts for metacontrol. In *Joint proceedings of the 40th International Computer Music Conference (ICMC), and the 11th Sound and Music Computing Conference (SMC)*. Athens, Greece.
- de Castro Lopo, Erik. (2020). Libsndfile. Retrieved April 20, 2020, from <http://www.mega-nerd.com/libsndfile/>
- De Poli, G. (2004). Methodologies for expressiveness modelling of and for music performance. *Journal of New Music Research*, 33(3), 189–202.
- de Cheveigné, A. & Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4), 1917–1930.
- Depalle, P., Garcia, G., & Rodet, X. (1993). Tracking of partials for additive sound synthesis using hidden Markov models. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processin (ICASSP)* (Vol. 1, 225–228 vol.1). Minneapolis, Minnesota, USA.

REFERENCES

- Desainte-Catherine, M. & Hanna, P. (2000). Statistical approach for sound modeling. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx)* (pp. 91–96). Verona, Italy.
- Deutsch, R. (1978). Constant speed portamento. US Patent 4,103,581.
- Devroye, L. (1986). *Non-uniform random variate generation*. McGill University: Springer.
- Di Donato, B. & Bullock, J. (2015). Gspat: Live sound spatialisation using gestural control. In *Student Think Tank at the 21st International Conference on Auditory Display* (p. 7). Graz, Austria.
- Di Donato, B., Bullock, J., & Tanaka, A. (2018). Myo mapper: A myo armband to OSC mapper. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Virginia Tech.
- Dirogis, P., Warusfel, O., & Caussi, R. (1996). Reproduction of directivity pattern using multi-loudspeaker source. *The Journal of the Acoustical Society of America*, 99(4), 2502–2529.
- Drugman, T. & Dutoit, T. (2012). The deterministic plus stochastic model of the residual signal and its applications. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(3), 968–981.
- Dubnov, S., Bar-Joseph, Z., El-Yaniv, R., Lischinski, D., & Werman, M. (2002). Synthesizing sound textures through wavelet tree learning. *IEEE Computer Graphics and Applications*, 22(4), 38–48.
- Dudley, H. (1939). Remaking speech. *The Journal of the Acoustical Society of America*, 11(2), 169–177.
- East West Communications, Inc. (2020, August 29). Quantum leap. Retrieved from <http://www.soundsonline.com/>
- Eerola, T. & Ferrer, R. (2008). Instrument library (MUMS) revised. *Music Perception: An Interdisciplinary Journal*, 25(3), 253–255.
- Eerola, T., Friberg, A., & Bresin, R. (2013). Emotional expression in music: Contribution, linearity, and additivity of primary musical cues. *Frontiers in Psychology*, 4, 487.
- Einbond, A. & Schwarz, D. (2010). Spatializing timbre with corpus-based concatenative synthesis. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 1–1). New York, United States.

REFERENCES

- Engel, J., Resnick, C., Roberts, A., Dieleman, S., Norouzi, M., Eck, D., & Simonyan, K. (2017). Neural audio synthesis of musical notes with wavenet autoencoders. In *Proceedings of the 34th International Conference on Machine Learning* (Vol. 70, pp. 1068–1077).
- Eom, H.-Y., Hempton, S., Huang, T.-S., Koo, H., Lamb, R., Ling, G., ... Xu, N. (2018). Keyboard instrument. US Patent D810 , 185.
- Erro, D., Sainz, I., Navas, E., & Hernaez, I. (2014). Harmonics plus noise model based vocoder for statistical parametric speech synthesis. *IEEE Journal of Selected Topics in Signal Processing*, 8(2), 184–194.
- Farnell, A. (2010). *Designing sound*. MIT Press.
- Favreau, E., Fingerhut, M., Koechlin, O., Potacsek, P., Puckette, M. S., & Rowe, R. (1986). Software developments for the 4x real-time system. In *Proceedings of the International Computer Music Conference (ICMC)*. Den Haag, The Netherlands.
- Fiebrink, R., Trueman, D., & Cook, P. R. (2009). A meta-instrument for interactive, on-the-fly machine learning. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 280–285). Pittsburgh, PA, USA.
- Fletcher, N. H. (2001). Vibrato in music. *Acoustics Australia*, 29(3).
- Fletcher, N. H. (2010). Vibrato in music - physics and psychophysics. In *Proceedings of the International Symposium on Music Acoustics*. Sydney - Katoomba, Australia.
- Fohl, W. & Nogalski, M. (2013). A gesture control interface for a wave field synthesis system. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 341–346). Daejeon, Korea.
- Freed, A. (1998). Real-time inverse transform additive synthesis for additive and pitch synchronous noise and sound spatialization. In *Proceedings of the 105th Audio Engineering Society Convention*. San Francisco, CA, USA.
- Friberg, A., Schoonderwaldt, E., & Juslin, P. N. (2007). CUEx: An algorithm for automatic extraction of expressive tone parameters in music performance from acoustic signals. *Acta Acustica united with Acustica*, 93(3), 411–420.
- Frigo, M. & Johnson, S. G. (2020). FFTW Fastest Fourier Transform in the West. Retrieved April 20, 2020, from <http://www.fftw.org/>
- Gabrielsson, A. & Juslin, P. N. (1996). Emotional expression in music performance: Between the performer's intention and the listener's experience. *Psychology of Music*, 24(1), 68–91.

REFERENCES

- Galamian, I. (1999). *Principles of violin playing and teaching*. Dover Books on Music.
- Garcia, J., Carpentier, T., & Bresson, J. (2017). Interactive-compositional authoring of sound spatialization. *Journal of New Music Research*, 46(1), 74–86.
- Gelineck, S. & Böttcher, N. (2012). 6to6Mappr: An educational tool for fast and easy mapping of input devices to musical parameters. In *Proceedings of the 7th Audio Mostly Conference: A Conference on Interaction with Sound* (pp. 117–123). Corfu, Greece.
- Genki Instruments. (2020, September 11). Genki Wave. Retrieved from <https://www.genkiinstruments.com/>
- George, E. B. & Smith, M. J. (1992). Analysis-by-synthesis/overlap-add sinusoidal modeling applied to the analysis and synthesis of musical tones. *Journal of the Audio Engineering Society*, 40(6), 497–516.
- Gertich, F., Gerlach, J., & Föllmer, G. (1996). *Musik, Verwandelt: das Elektronische Studio der TU Berlin 1953-1995*. Books on Music. Wolke Verlag.
- Goebel, W., Dixon, S., De Poli, G. D., Friberg, A., & Bresin, R. (2008). Sense in expressive music performance: Data acquisition, computational studies, and models. In *Sound to sense - sense to sound: A state of the art in sound and music computing* (pp. 195–242). Logos Verlag.
- Goodwin, M. [M.]. (1996). Residual modeling in music analysis-synthesis. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (Vol. 2, 1005–1008 vol. 2). Atlanta, Georgia, USA.
- Goodwin, M. & Rodet, X. (1994). Efficient Fourier synthesis of nonstationary sinusoids. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 333–334). Aarhus, Denmark.
- Goto, M. et al. (2004). Development of the RWC music database. In *Proceedings of the 18th International Congress on Acoustics (ICA)* (Vol. 1, pp. 553–556). Kyoto, Japan.
- Goto, M., Hashiguchi, H., Nishimura, T., & Oka, R. (2002). RWC music database: Popular, classical and jazz music databases. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)* (Vol. 2, pp. 287–288). Paris, France.
- Götz, P. M. (2018). *Analysis and synthesis of control parameters in note transitions* (Master's thesis, TU Berlin).

REFERENCES

- Grani, F., Overholt, D., Erkut, C., Gelineck, S., Triantafyllidis, G., Nordahl, R., & Serafin, S. (2015). Spatial sound and multimodal interaction in immersive environments. In *Proceedings of the Audio Mostly Conference on Interaction With Sound* (pp. 1–5). Thessaloniki, Greece.
- Guo, H. (2011). A simple algorithm for fitting a gaussian function [dsp tips and tricks]. *IEEE Signal Processing Magazine*, 28, 134–137.
- Gurevich, M. (2015). Interacting with Cage: Realising classic electronic works with contemporary technologies. *Organised Sound*, 20, 290–299.
- Hagan, K. L. (2017). Textural composition: Aesthetics, techniques, and spatialization for high-density loudspeaker arrays. *Computer Music Journal*, 41(1), 34–45.
- Hahn, H. (2015). *Expressive sampling synthesis-learning extended source-filter models from instrument sound databases for expressive sample manipulations* (Doctoral dissertation, UPMC Université Paris VI).
- Hahn, H. & Röbel, A. (2012). Extended source-filter model of quasi-harmonic instruments for sound synthesis, transformation and interpolation. In *Proceedings of the 9th Sound and Music Computing Conference (SMC)* (pp. 434–441). Copenhagen, Denmark.
- Hahn, H., Röbel, A., Burred, J. J., & Weinzierl, S. (2010). Source-filter model for quasi-harmonic instruments. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. Graz, Austria.
- Haken, L. (2004). Continuous music keyboard. US Patent 6,703,552.
- Haken, L., Fitz, K., & Christensen, P. (2007). Beyond traditional sampling synthesis: Real-time timbre morphing using additive synthesis. In J. W. Beauchamp (Ed.), *Analysis, Synthesis and Perception of Musical Sounds* (pp. 122–144). Springer Verlag.
- Haken, L., Fitz, K., Tellman, E., Wolfe, P. J., & Christensen, P. (1997). A continuous music keyboard controlling polyphonic morphing using bandwidth-enhanced oscillators. In *Proceedings of the International Computer Music Conference (ICMC)*. Thessaloniki, Greece.
- Haken, L., Tellman, E., & Wolfe, P. (1998). An indiscrete music keyboard. *Computer Music Journal*, 22(1), 30–48.
- Haller, H. P. (1995). *Das Experimentalstudio der Heinrich-Strobel-Stiftung des Südwestfunks Freiburg 1971-1989: die Erforschung der elektronischen Klangumformung und ihre Geschichte, Teil 1*. Baden-Baden: Nomos.

REFERENCES

- Hamon, C., Mouline, E., & Charpentier, F. (1989). A diphone synthesis system based on time-domain prosodic modifications of speech. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (238–241 vol.1). Glasgow, Scotland.
- Harris, S. & Sinclair, S. (2020). Lightweight OSC implementation: Git repository. Retrieved April 20, 2020, from <https://github.com/radarsat1/liblo>
- Harrison, J. (1998). Sound, space, sculpture: Some thoughts on the ‘what’, ‘how’ and ‘why’ of sound diffusion. *Organised Sound*, 3(2), 117–127.
- Hartvigsen, D. (2014). Fingering systems for electronic musical instruments. *Journal of Mathematics and Music*, 8(1), 41–58.
- Hauck, W. (2000). *Das Vibrato auf der Violine*. Bosworth Music.
- Hayafuchi, K. & Suzuki, K. (2008). MusicGlove: A wearable musical controller for massive media library. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Genova, Italy.
- Hellenkemper, N. (2007). *Instrumentalvibrato im 19. Jahrhundert: Technik, Anwendung, Notationsformen : mit einem Ausblick ins 20. Jahrhundert*. Schriften zur Musikwissenschaft aus Münster. K.D. Wagner.
- Henriques, T. (2008). META-EVI innovative performance paths with a wind controller. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 307–310). Genova, Italy.
- Herrera, P. & Bonada, J. (1998). Vibrato extraction and parameterization in the spectral modeling synthesis framework. In *Proceedings of the Digital Audio Effects Workshop (DAFX)* (pp. 107–110). Barcelona, Spain.
- Hoelzl, H., Han, I., & de Campo, A. (2019). The airborne instruments nufo: A movement based musical instrument for possibility space exploration. In *Proceedings of the 6th International Conference on Movement and Computing* (pp. 1–4). Tempe, Arizona.
- Hope, C. A. & James, S. (2013). 2D AND 3D timbral spatialisation: Spatial motion, immersiveness, and notions of space. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 77–84). Perth, Australia.
- Hunt, A. & Kirk, R. (2000). Mapping strategies for musical performance. *Trends in Gestural Control of Music*, 21(2000), 231–258.

REFERENCES

- Hunt, A. & Wanderley, M. M. (2002). Mapping performer parameters to synthesis engines. *Organised Sound*, 7(2), 97–108.
- James, S. (2012). From autonomous to performative control of timbral spatialisation. In *Proceedings of the Australasian Computer Music Conference* (pp. 31–38). Griffith University, Australia.
- James, S. (2015). Spectromorphology and spatiomorphology of sound shapes: Audio-rate AEP and DBAP panning of spectra. In *Proceedings of the International Computer Music Conference (ICMC)*. Denton, Texas, United States.
- Janer, J., Bonada, J., & Blaauw, M. (2006). Performance-driven control for sample-based singing voice synthesis. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. Montreal, Canada.
- Jenkins, M. (2019). *Analog synthesizers: Understanding, performing, buying: From the legacy of Moog to software synthesis*. Routledge.
- Johnson, B. & Kapur, A. (2013). Multi-touch interfaces for phantom source positioning in live sound diffusion. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 213–216). Daejeon + Seoul, Korea.
- Johnson, B., Norris, M., & Kapur, A. (2014). Diffusing diffusion: A history of the technological advances in spatial performance. In *Proceedings of the International Computer Music Conference (ICMC)*. Athens, Greece.
- Juslin, P. N. (2003). Five facets of musical expression: A psychologist's perspective on music performance. *Psychology of Music*, 31(3), 273–302.
- Karplus, K. & Strong, A. (1983). Digital synthesis of plucked-string and drum timbres. *Computer Music Journal*, 7(2), 43–55.
- Kendall, G. S. (2010). Spatial perception and cognition in multichannel audio for electroacoustic music. *Organised Sound*, 15(3), 228–238.
- Kendall, G. S. & Ardila, M. (2008). The artistic play of spatial organization: Spatial attributes, scene analysis and auditory spatial schemata. In *Computer music modeling and retrieval: Sense of sounds* (pp. 125–138). Springer.
- Kestelli, S. S. (2019). *Exploring hand interfaces within new digital musical instruments research* (Doctoral dissertation, Istanbul Technical University, Department of Music).
- Kim-Boyle, D. (2005). Sound spatialization with particle systems. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Madrid, Spain.

REFERENCES

- Kim-Boyle, D. (2008). Spectral spatialization - an overview. In *Proceedings of the International Computer Music Conference (ICMC)*. Belfast, UK.
- Kitware Inc. (2020, October 3). CMake Documentation. Retrieved from <https://cmake.org/cmake/help/v3.18/>
- Klapuri, A. (2007). Analysis of musical instrument sounds by source-filter-decay model. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 53–56). Honolulu, Hawaii.
- Lagrange, M., Marchand, S., & Rault, J.-B. (2004). Using linear prediction to enhance the tracking of partials. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (Vol. 4). Montreal, Quebec, Canada.
- Lagrange, M., Marchand, S., & Rault, J.-B. (2007). Enhancing the tracking of partials for the sinusoidal modeling of polyphonic sounds. *IEEE Transactions on Audio, Speech, and Language Processing*, 15, 1625–1634.
- Lagrange, M., Marchand, S., & Rault, J.-B. (2005). Tracking partials for the sinusoidal modeling of polyphonic sounds. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (Vol. 3). Philadelphia, Pennsylvania, USA.
- Laroche, J. (2000). Synthesis sinusoids via non-overlapping inverse fourier transform. *IEEE Transactions on Speech and Audio Processing*, 8(4), 471–477.
- Laroche, J., Stylianou, Y., & Moulines, E. (1993). HNS: Speech modification based on a harmonic+noise model. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (Vol. 2, pp. 550–553). Minneapolis, Minnesota, USA.
- Laugwitz, B., Held, T., & Schrepp, M. (2008). Construction and evaluation of a user experience questionnaire. In *Proceedings of the 4th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering of the Austrian Computer Society* (Vol. 5298, pp. 63–76). Graz, Austria.
- Laurens, H. (1957). Electronic oscillator. US Patent 2,790,906.
- Lee, M., Freed, A., & Wessel, D. (1991). Real-time neural network processing of gestural and acoustic signals. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 277–277). Montreal, Quebec, Canada.
- Lerch, A. (2012). *An introduction to audio content analysis: Applications in signal processing and music informatics*. John Wiley & Sons.

REFERENCES

- Lerdahl, F. (2009). Genesis and architecture of the GTTM project. *Music Perception: An Interdisciplinary Journal*, 26(3), 187–194.
- Leslie, G., Zamborlin, B., Jodlowski, P., & Schnell, N. (2010). Grainstick: A collaborative, interactive sound installation. In *Proceedings of the International Computer Music Conference (ICMC)* (p. 4). New York, USA.
- Levine, S. & Smith, J. (1998). A sines + transients + noise audio representation for data compression and time/pitch scale modifications. In *Proceedings of the 105th Audio Engineering Society Convention*. San Francisco, CA.
- Levitin, D. J., McAdams, S., & Adams, R. L. (2002). Control parameters for musical instruments: A foundation for new mappings of gesture to sound. *Organised Sound*, 7(2), 171–189.
- Lindau, A. (2015). *Spatial audio quality inventory (SAQI). Test manual. v1.2*. TU Berlin.
- Lindau, A., Erbes, V., Lepa, S., Maempel, H.-J., Brinkman, F., & Weinzierl, S. (2014). A spatial audio quality inventory (SAQI). *Acta Acustica united with Acustica*, 100(5), 984–994.
- Lindemann, E. (2001). Musical synthesizer capable of expressive phrasing. US Patent: 6316710.
- Lindemann, E. (2007). Music Synthesis with Reconstructive Phrase Modeling. *IEEE Signal Processing Magazine*, 24, 80–91.
- Lindemann, E. (2010). Sound synthesis by combining a slowly varying underlying spectrum, pitch and loudness with quicker varying spectral, pitch and loudness fluctuations. US Patent: 5744742.
- Lindemann, E. (2011). Synful website. Retrieved June 28, 2020, from <https://www.synful.com>
- Linn, R. (2020, August 14). Linnstrument. Retrieved from <https://www.rogerlinndesign.com/linnstrument>
- London, J. (2000). Musical expression and musical meaning in context. In *Proceedings of the 6th International Conference on Music Perception and Cognition*. Keele, UK.
- Loureiro, M., Yehia, H., de Paula, H., Campolina, T., & Mota, D. (2009). Content analysis of note transitions in music performance. In *Proceedings of the 6th Sound and Music Computing Conference (SMC)*. Porto, Portugal.
- Lynch, H. & Sazdov, R. (2011). An ecologically valid experiment for the comparison of established spatial techniques. In *Proceedings of the International Computer Music Conference (ICMC)*. Huddersfield, UK.

REFERENCES

- Machover, T. (1992). *Hyperinstruments: A progress report, 1987-1991*. MIT Media Laboratory.
- Mackay, W. (2000). Responding to cognitive overload: Co-adaptation between users and technology. *Intellectica*, 30(1), 177–193.
- Mackinlay, J., Card, S. K., & Robertson, G. G. (1990). A semantic analysis of the design space of input devices. *Human-Computer Interaction*, 5(2), 145–190.
- Madronalabs. (2020, August 14). Soundplane. Retrieved from <https://madronalabs.com/soundplane>
- Maestre, E., Bonada, J., & Mayor, O. (2006). Modeling musical articulation gestures in singing voice performances. In *Proceedings of the 121st Audio Engineering Society Convention*. San Francisco, CA, USA.
- Maestre, E. & Gomez, E. (2005). Automatic characterization of dynamics and articulation of expressive monophone recordings. In *Proceedings of the 118th Audio Engineering Society Convention*. Barcelona, Spain.
- Maestre, E., Ramirez, R., Kersten, S., & Serra, X. (2009). Expressive concatenative synthesis by reusing samples from real performance recordings. *Computer Music Journal*, 33, 23–42.
- Malloch, J., Garcia, J., Wanderley, M. M., Mackay, W. E., Beaudouin-Lafon, M., & Huot, S. (2019). A design workbench for interactive music systems. In *New Directions in Music and Human-Computer Interaction* (pp. 23–40)
- Malloch, J., Sinclair, S., & Wanderley, M. M. (2013). Libmapper (a library for connecting things). In *CHI'13 Extended Abstracts on Human Factors in Computing Systems* (pp. 3087–3090). Paris, France.
- Manaris, B. & Stoudenmier, S. (2015). Specter: Combining music information retrieval with sound spatialization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)* (Vol. 289). Malaga, Spain.
- Marchand, S. (2012). The simplest analysis method for non-stationary sinusoidal modeling. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)* (pp. 23–26). York, United Kingdom.
- Marentakis, G., Peters, N., & McAdams, S. (2007). Dj SPAT: Spatialized interactions for DJs. In *Proceedings of the International Computer Music Conference (ICMC)* (Vol. 2, pp. 89–94). San Francisco, USA.

REFERENCES

- Marshall, M. T., Hartshorn, M., Wanderley, M. M., & Levitin, D. J. (2009). Sensor choice for parameter modulations in digital musical instruments: Empirical evidence from pitch modulation. *Journal of New Music Research*, 38(3), 241–253.
- Marshall, M., Malloch, J., & Wanderley, M. (2007). Gesture control of sound spatialization for live musical performance. In *Proceedings of the 7th International Gesture Workshop - Gesture-Based Human-Computer Interaction and Simulation* (Vol. 5085, pp. 227–238). Lisbon, Portugal.
- Marshall, M., Peters, N., Jensenius, A. R., Boissinot, J., Wanderley, M. M., & Braasch, J. (2006). On the development of a system for gesture control of spatialization. In *Proceedings of the International Computer Music Conference (ICMC)*. New Orleans, USA.
- Master, A. S. & Liu, Y.-W. (2003). Nonstationary sinusoidal modeling with efficient estimation of linear frequency chirp parameters. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (Vol. 5). Hong Kong, China.
- Mathews, M. V. (1969). *The Technology of Computer Music*. MIT Press.
- McAulay, R. & Quatieri, T. (1986). Speech analysis/synthesis based on a sinusoidal representation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(4), 744–754.
- McGee, R. (2015). Spatial modulation synthesis. In *Proceedings of the International Computer Music Conference (ICMC)*. Denton, USA.
- McGuire, S. (2007). *Software sampling: A practical guide*. Elsevier.
- McLeran, A., Roads, C., Sturm, B. L., & Shynk, J. J. (2008). Granular sound spatialization using dictionary-based methods. In *Proceedings of the 5th Sound and Music Computing Conference (SMC)*. Berlin, Germany.
- McPherson, A. (2012). Touchkeys: Capacitive multi-touch sensing on a physical keyboard. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Ann Arbor, Michigan, USA.
- Mellody, M. & Wakefield, G. H. (2000). The time-frequency characteristics of violin vibrato: Modal distribution analysis and synthesis. *Journal of the Acoustical Society of America*, 107, 598–611.
- Meurisse, G., Hanna, P., & Marchand, S. (2006). A new analysis method for sinusoids+noise spectral models. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)* (pp. 139–144). Montreal, Canada.

REFERENCES

- Meyer, J. (2008). Musikalische Akustik. In S. Weinzierl (Ed.), *Handbuch der Audiotechnik* (pp. 123–180). VDI-Buch. Berlin Heidelberg: Springer.
- Misdariis, N., Warusfel, O., & Causse, R. (2001). Radiation control on a multi-loudspeaker device. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 306–309). Havana, Cuba.
- Mitchell, H. F. & Kenny, D. T. (2009). Change in vibrato rate and extent during tertiary training in classical singing students. *Journal of Voice*, (24), 427–34.
- Mitchell, T. (2011). Soundgrasp: A gestural interface for the performance of live music. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Oslo, Norway.
- Müllensiefen, D., Gingras, B., Musil, J., & Stewart, L. (2014). Measuring the facets of musicality: The goldsmiths musical sophistication index (Gold-MSI). *Personality and Individual Differences*, 60, S35.
- Müllensiefen, D., Gingras, B., Musil, J., & Stewart, L. (2014). The musicality of non-musicians: An index for assessing musical sophistication in the general population. *PLOS ONE*, 9(2), 1–23
- Müller, A. & Rabenstein, R. (2009). Physical modeling for spatial sound synthesis. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Como, Italy.
- Müller, J., Geier, M., Dicke, C., & Spors, S. (2014). The BoomRoom: Mid-air direct interaction with virtual sound sources. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 247–256). Toronto, Ontario, Canada.
- Naef, M. & Collicott, D. (2006). A VR interface for collaborative 3d audio performance. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 57–60). Paris, France.
- Native Instruments. (2020, August 29). Third-party sample libraries for Kontakt. Retrieved from <https://www.native-instruments.com/de/products/komplete/samplers/kontakt-6-player/third-party-sample-libraries/>
- Nienhuys, H.-W. & Nieuwenhuizen, J. (2003). LilyPond, a system for automated music engraving. In *Proceedings of the XIV Colloquium on Musical Informatics (XIV CIM)* (Vol. 1, pp. 167–171). Firenze, Italy.
- Nikitin, P. (2012). Leon Theremin (Lev Termen). *IEEE Antennas and Propagation Magazine*, 54(5), 252–257.

REFERENCES

- Norman, Alex. (2012). JackCpp: Git Repository. Retrieved April 20, 2020, from <https://github.com/x37v/jackcpp>
- Normandeau, R. (2009). Timbre spatialisation: The medium is the space. *Organised Sound*, 14(3), 277–285.
- Nyström, E. (2015). Low-level topology of spatial texture. In *Proceedings of the International Computer Music Conference (ICMC)*. University of North Texas, USA.
- Nyström, E. (2018). Topographic synthesis: Parameter distribution in spatial texture. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 117–122). ICMC. Daegu, Korea.
- Olson, H. F. & Belar, H. (1955). Electronic music synthesizer. *The Journal of the Acoustical Society of America*, 27(3), 595–612.
- Oord, A. v. d., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., . . . Stimberg, F., et al. (2017). Parallel WaveNet: Fast high-fidelity speech synthesis. In *Proceedings of the 35th International Conference on Machine Learning*. Stockholm, Sweden.
- Oppenheim, D. V. & Wright, J. (1996). Towards a framework for handling musical expression. In *Proceedings of the International Computer Music Conference (ICMC)*. Hong Kong.
- Palmer, C. & Hutchins, S. (2006). What is musical prosody? *The Psychology of Learning and Motivation: Advances in Research and Theory*, 46, 245–278.
- Pang, H.-S. & Yoon, D.-H. (2005). Automatic detection of vibrato in monophonic music. *Pattern Recognition*, 38(7), 1135–1138.
- Pantazis, Y. & Stylianou, Y. (2008). Improving the modeling of the noise part in the harmonic plus noise model of speech. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 4609–4612). Las Vegas, Nevada, USA.
- Paradiso, J. A. (1997). Electronic music: New ways to play. *IEEE Spectrum*, 34(12), 18–30.
- Perez-Lopez, A. (2015). 3Dj: A SuperCollider framework for real-time sound spatialization. In *Proceedings of the 21th International Conference on Auditory Display*. Graz, Austria.
- Perez, A., Bonada, J., Maestre, E., Guaus, E., & Blaauw, M. (2007). Combining performance actions with spectral models for violin sound transformation. In *International Congress on Acoustics*. Madrid, Spain.

REFERENCES

- Peters, N. (2011). *Sweet [re] production: Developing sound spatialization tools for musical applications with emphasis on sweet spot and off-center perception* (Doctoral dissertation, McGill University, Montreal, Canada).
- Philharmonia Orchestra London. (2020, February 27). Philharmonia sound samples. Retrieved from <https://philharmonia.co.uk/resources/sound-samples/>
- Pihlajamäki, T., Santala, O., & Pulkki, V. (2014). Synthesis of spatially extended virtual source with time-frequency decomposition of mono signals. *Journal of the Audio Engineering Society*, 62(7/8), 467–484.
- Pinch, T. (2015). Between technology and music. In R. Garud, B. Simpson, A. Langley, & H. Tsoukas (Eds.), *The emergence of novelty in organizations* (Vol. 5, p. 129). Oxford University Press, USA.
- Prame, E. (1994). Measurements of the vibrato rate of ten singers. *Journal of the Acoustical Society of America*, 96(4), 1979–1984.
- Puckette, M. S. (1988). The patcher. In *Proceedings of the International Computer Music Conference (ICMC)*. Cologne, Germany.
- Puckette, M. S. (1997). Pure Data. In *Proceedings of the International Computer Music Conference (ICMC)*. Thessaloniki, Greece.
- Pulkki, V., Laitinen, M.-V., & Erku, C. (2009). Efficient spatial sound synthesis for virtual worlds. In *Proceedings of the 35th International Audio Engineering Society Conference on Audio for Games*. London, UK.
- Pysiewicz, A. & Weinzierl, S. (2017). Instruments for spatial sound control in real time music performances. a review. In *Musical instruments in the 21st century* (pp. 273–296). Springer.
- Quartier, L., Meurisse, T., Colmars, J., Frelat, J., & Vaiedelich, S. (2015). Intensity key of the Ondes Martenot: An early mechanical haptic device. *Acta Acustica united with Acustica*, 101(2), 421–428.
- Quatieri, T. & McAulay, R. (1986). Speech transformations based on a sinusoidal representation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(6), 1449–1464.
- Rabenstein, R. (2010). Spatial sound synthesis for circular membranes. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Graz, Austria.
- Rabenstein, R. & Trautmann, L. (2003). Digital sound synthesis of string instruments with the functional transformation method. *IEEE Signal Processing Magazine*, 83(8), 1673–1688.

REFERENCES

- Ratcliffe, J. (2014). Hand motion-controlled audio mixing interface. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. London, UK.
- Regnier, L. & Peeters, G. (2009). Singing voice detection in music tracks using direct voice vibrato detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Taipei, Taiwan.
- Roads, C. (1978). Automated Granular Synthesis of Sound. *Computer Music Journal*, 2(2), 61–62.
- Roads, C. (2004). *Microsound*. The MIT Press.
- Roads, C. (2005). The art of articulation: The electroacoustic music of Horacio Vaggione. *Contemporary Music Review*, 24(4-5), 295–309.
- Roads, C. (2006). The evolution of granular synthesis: An overview of current research. In *International Symposium on The Creative and Scientific Legacies of Iannis Xenakis*. Guelph/Waterloo/Toronto, Canada.
- Röbel, A. (2008). Frequency-slope estimation and its application to parameter estimation for non-stationary sinusoids. *Computer Music Journal*, 32(2), 68–79.
- Rodet, X. & Depalle, P. (1992). Spectral envelopes and inverse FFT synthesis. In *Proceedings of the Audio Engineering Society Convention*. San Francisco, USA.
- Rodet, X. & Schwarz, D. (2007). Spectral envelopes and additive + residual analysis/synthesis. In J. Beauchamp (Ed.), *Analysis, synthesis, and perception of musical sounds* (pp. 175–227). Modern Acoustics and Signal Processing. Springer New York.
- Roli Ltd. (2020, August 14). Seaboard. Retrieved from <https://roli.com/products/seaboard>
- Rosli, M. H. W. & Roads, C. (2016). Spatiotemporal granulation. In *Proceedings of the International Computer Music Conference (ICMC)*. Utrecht, Netherlands.
- Ross, M. P. (2007). *Multi-observation continuous density hidden markov models for anomaly detection in full motion video* (Doctoral dissertation, Air Force Institute of Technology).
- Rossignol, S. (2000). *Segmentation et indexation des signaux sonores musicaux* (Doctoral dissertation, IRCAM, Paris).
- Rossignol, S., Depalle, P., Soumagne, J., Rodet, X., & Collette, J.-L. (1999). Vibrato: Detection, estimation, extraction, modification. In *Digital Audio Effects Workshop (DAFx)* (pp. 175–179). Trondheim, Norway.

REFERENCES

- Rossignol, S., Rodet, X., Soumagne, J., Colette, J.-L., & Depalle, P. (1998). Feature extraction and temporal segmentation of acoustic signals. In *Proceedings of the International Computer Music Conference (ICMC)*. Ann Arbor, Michigan, USA.
- Rovan, J. B., Wanderley, M. M., Dubnov, S., & Depalle, P. (1997). Instrumental gestural mapping strategies as expressivity determinants in computer music performance. In *Proceedings of the AIMI International Workshop* (pp. 68–73). Genova, Italy.
- Saino, K., Tachibana, M., & Kenmochi, H. (2010). A singing style modeling system for singing voice synthesizers. In *11th Annual Conference of the International Speech Communication Association*. Makuhari, Chiba, Japan.
- Saito, Y., Takamichi, S., & Saruwatari, H. (2017). Statistical parametric speech synthesis incorporating generative adversarial networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(1), 84–96.
- Saitou, T., Unoki, M., & Akagi, M. (2002). Extraction of F0 dynamic characteristics and development of F0 control model in singing voice. In *Proceedings of the International Conference on Auditory Display*. Kyoto, Japan.
- Sala, O. (1950). Das Mixtur-Trautonium. *Physikalische Blätter*, 6(9), 390–398.
- Sasamoto, Y., Cohen, M., & Villegas, J. (2013). Controlling spatial sound with table-top interface. In *Proceedings of the International Joint Conference on Awareness Science and Technology* (pp. 713–718). Aizuwakamatsu, Japan.
- Schacher, J. C. (2007). Gesture control of sounds in 3D space. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 358–362). New York, USA.
- Schaeffer, P. (2012). *In Search of a Concrete Music*. California Studies in 20th-Century Music. Translated by C. North and J. Dack. University of California Press.
- Schedel, M. (2007). Electronic music and the studio. In N. Collins (Ed.), *The cambridge companion to electronic music* (pp. 24–37).
- Schmele, T. (2011). *Exploring 3d audio as a new musical language* (Master's thesis, Universitat Pompeu Fabra).
- Schmied, A. (2018). *Developement and stress testing of the BinBong MKII* (Master's thesis, TU Berlin, Berlin, Germany).
- Schumacher, M. & Bresson, J. (2010). Spatial sound synthesis in computer-aided composition. *Organised Sound*, 15(3), 271–289.

REFERENCES

- Schwarz, D. (2000). A system for data-driven concatenative sound synthesis. In *Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx)*. Verona, Italy.
- Schwarz, D. (2004). *Data-driven concatenative sound synthesis* (Doctoral dissertation, Université Paris 6 - Pierre et Marie Curie).
- Schwarz, D., Liu, W., & Bevilacqua, F. (2020). A survey on the use of 2D touch interfaces for musical expression. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Birmingham, UK (online).
- Seashore, C. E. (1938). *Psychology of Music*. Dover.
- Serafin, S., Trento, S., Grani, F., Perner-Wilson, H., Madgwick, S., & Mitchell, T. J. (2014). Controlling physically based virtual musical instruments using the Gloves. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. London, UK.
- Serra, X. (1989). *A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition* (Doctoral dissertation, Stanford University).
- Serra, X. (1997). Musical sound modeling with sinusoids plus noise. In C. Roads, S. T. Pope, A. Piccialli, & G. D. Poli (Eds.), *Musical signal processing* (pp. 91–122). Lisse, the Netherlands.
- Serra, X. & Smith, J. (1990). Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition. *Computer Music Journal*, 14(4), 12–14.
- Smalley, D. (1997). Spectromorphology: explaining sound-shapes. *Organised Sound*, 2(02), 107–126.
- Smith, J. S., Julius o. an Abel. (1999). Bark and ERB bilinear transforms. *IEEE Transactions on Speech and Audio Processing*, 7(6), 697–708.
- Smith, J. O. (1991). Viewpoints on the history of digital synthesis. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 1–10). Montreal, Quebec, Canada.
- Smith, J. O. & Serra, X. (1987). PARSHL: an analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 290–297). Urbana, Illinois, USA.
- Solomos, M. (2006). The granular connection (Xenakis, Vaggione, Di Scipio...) In *Symposium On The Creative and Scientific Legacies of Iannis Xenakis*. Guelph/Waterloo/Toronto, Canada.
- Sonami, L. (2020, August 17). Lady's Glove. Retrieved from https://sonami.net/?page_id=118

REFERENCES

- Springer, J. P., Sladeczek, C., Scheffler, M., Hochstrate, J., Melchior, F., & Frohlich, B. (2006). Combining wave field synthesis and multi-viewer stereo displays. In *Proceedings of the IEEE Virtual Reality Conference (VR 2006)* (pp. 237–240). IEEE. Alexandria, VA, USA.
- Staatliches Institut für Musikforschung. (2020, March 1). Reflexionsarmer Raum. Retrieved from https://www.sim.spk-berlin.de/refelxionsarmer_raum_544.html
- Stockhausen, K. (1963). Texte zur elektronischen und instrumentalen Musik: Aufsätze 1952-1962 zur Theorie des Komponierens. (Chap. Arbeitsbericht 1953: Die Entstehung der Elektronischen Musik, Vol. 1, p. 42).
- Stockhausen, K. (1964). *Texte zur elektronischen und instrumentalen Musik; Band 2: Texte zu eigenen Werken zur Kunst Anderer, Aktuelles*. DuMont Schauberg.
- Stockhausen, K. (1971). *Texte zur Musik 1963-1970, Band 3: Einführungen und Projekte, Kurse, Sendungen, Standpunkte, Nebennoten*. DuMont Schauberg.
- Stockhausen, K. (1989). *Ypsilon, for a melody instrument (with micro-tones)*. Kürten, Germany: Stockhausen-Verlag,
- Strawn, J. (1985). *Modeling musical transitions* (Ph.D. CCRMA, Department of Music, Stanford University).
- Strawn, J. (1986). Orchestral instruments: Analysis of performed transitions. *Journal of the Audio Engineering Society*, 34(11), 867–880.
- Stylianou, Y. (2001). Applying the harmonic plus noise model in concatenative speech synthesis. *IEEE Transactions on Speech and Audio Processing*, 9(1), 21–29.
- Tahiroğlu, K. (2011). An exploration on mobile interfaces with adaptive mapping strategies in pure data. In *Proceedings of the Pure Data Convention*. Bauhaus-Universität Weimar, Germany.
- Tanaka, A., Di Donato, B., Zbyszynski, M., & Roks, G. (2019). Designing gestures for continuous sonic interaction. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Porto Alegre, Brazil.
- Tolonen, T., Välimäki, V., & Karjalainen, M. (1998). *Evaluation of modern sound synthesis methods*. Helsinki University of Technology. Espoo.
- Topper, D., Burtner, M., & Serafin, S. (2002). Spatio-operational spectral (SOS) synthesis. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Hamburg, Germany.

REFERENCES

- Torchia, R. H. & Lippe, C. (2004). Techniques for multi-channel real-time spatial distribution using frequency-domain processing. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 116–119). National University of Singapore.
- Torre, G., Andersen, K., & Baldé, F. (2016). The Hands: The making of a digital musical instrument. *Computer Music Journal*, 40(2), 22–34.
- Treindl, G. (2016). *Entwicklung und Evaluation eines Controllers zur Tonhöhensteuerung über Vier-Finger-Kombinationen* (Master's thesis, TU Berlin).
- Truax, B. (1987). Real-time granulation of sampled sound with the dmx-1000. In *Proceedings of the International Computer Music Conference (ICMC)*. Champaign/Urbana, Illinois, USA.
- Trueman, D. & Cook, P. (2000). Bossa: The deconstructed violin reconstructed. *Journal of New Music Research*, 29(2), 121–130.
- Valle, A., Tazelaar, K., & Lombardo, V. (2010). In a concrete space: Reconstructing the spatialization of Iannis Xenakis' Concret PH on a multichannel setup. In *Proceedings of the Sound and Music Computing Conference (SMC)*. Copenhagen, Denmark.
- Verfaillie, V., Guastavino, C., & Depalle, P. (2005). Perceptual evaluation of vibrato models. In *Proceedings of the Conference on Interdisciplinary Musicology (CIM)*. Montréal, Canada.
- Verfaillie, V., Wanderley, M. M., & Depalle, P. (2006). Mapping strategies for gestural and adaptive control of digital audio effects. *Journal of New Music Research*, 35(1), 71–93.
- Verma, T. S., Levine, S. N., & Meng, T. H. Y. (1997). Transient modeling synthesis: A flexible analysis/synthesis tool for transient signals. In *Proceedings of the International Computer Music Conference (ICMC)*. Thessaloniki, Greece.
- Verma, T. S. & Meng, T. H. Y. (1998). Time scale modification using a sines + transients + noise signal model. In *Proceedings of the Digital Audio Effects Workshop (DAFx)* (pp. 49–52). Barcelona, Spain.
- Verron, C., Aramaki, M., Kronland-Martinet, R., & Pallone, G. (2008). Spatialized additive synthesis of environmental sounds. In *Proceedings of the 125 Audio Engineering Society Convention*. San Francisco, USA.
- Verron, C., Aramaki, M., Kronland-Martinet, R., & Pallone, G. (2009). A 3-D immersive synthesizer for environmental sounds. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6), 1550–1561.

REFERENCES

- Verron, C., Pallone, G., Aramaki, M., & Kronland-Martinet, R. (2009). Controlling a spatialized environmental sound synthesizer. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics* (pp. 321–324). New Paltz, NY, USA.
- Vertegaal, R., Ungvary, T., & Kieslinger, M. (1996). Towards a musician’s cockpit: Transducers, feedback and musical function. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 308–311). Hong Kong, China.
- Vienna Symphonic Library GmbH. (2020, August 29). Vienna Symphonic Library. Retrieved from <https://www.vsl.co.at/en>
- Visi, F., Caramiaux, B., & Mcloughlin, M. (2017). A knowledge-based, data-driven method for action-sound mapping. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Copenhagen, Denmark.
- von Helmholtz, H. (1877). *On the Sensations of Tone*. Dover Publications.
- von Coler, H. (2013). *Blind note-level segmentation of monophonic music using hidden Markov models* (Master’s thesis, TU Berlin).
- von Coler, H. (2014). Musical expressivity - considerations between sound synthesis, musicology and other disciplines. In *Proceedings of the 9th Conference on Interdisciplinary Musicology - CIM14* (pp. 122–125). Berlin, Germany.
- von Coler, H. (2018). TU-Note Violin Sample Library – a database of violin sounds with segmentation ground truth. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Aveiro, Portugal.
- von Coler, H. (2019a). A JACK-based application for spectro-spatial additive synthesis. In *Proceedings of the 17th Linux Audio Conference (LAC-19)*. Stanford University, CA, USA.
- von Coler, H. (2019b). Statistical sinusoidal modeling for expressive sound synthesis. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Birmingham, UK.
- von Coler, H. (2020a, October 12). BinBong: Software repository. Retrieved from <https://github.com/anwaltdt/binbong>
- von Coler, H. (2020b, September 4). GLOOO analysis: Software repository. Retrieved from https://github.com/anwaltdt/GLOOO_analysis
- von Coler, H. (2020c, September 14). GLOOO synth: Software repository. Retrieved from https://github.com/anwaltdt/GLOOO_synth

REFERENCES

- von Coler, H. (2020d, September 4). GLOOO synth documentation: Source code documentation. Retrieved April 20, 2020, from https://hvc.berlin/glooo_synth_doc/index.html
- von Coler, H. (2020e, October 16). Python OSC sequencer: Software repository. Retrieved from <https://github.com/anwaldt/py-osc-seq>
- von Coler, H. (2020f, October 12). SC Spat: Software repository. Retrieved from https://github.com/anwaldt/SC_Spat
- von Coler, H., Götz, M., & Lepa, S. (2018). Parametric synthesis of glissando note transitions – a user study in a real-time application. In *Proceedings of the International Conference of Digital Audio Effects (DAFx)*. Aveiro, Portugal.
- von Coler, H., Lepa, S., & Weinzierl, S. (2020). User-defined mappings for spatial sound synthesis. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Birmingham, UK (Online).
- von Coler, H. & Lerch, A. (2014). CMMSD: A data set for note-level segmentation of monophonic music. In *Proceedings of the AES 53rd International Conference on Semantic Audio*. London, UK.
- von Coler, H., Margraf, J., & Schuladen, P. (2018). TU-Note Violin Sample Library. <http://dx.doi.org/10.14279/depositonce-6747>. Data set. TU-Berlin.
- von Coler, H. & Röbel, A. (2011). Vibrato detection using cross correlation between temporal energy and fundamental frequency. In *Proceedings of the 131st Audio Engineering Society Convention*. New York, USA.
- von Coler, H., Treindl, G., Egermann, H., & Weinzierl, S. (2017). Development and evaluation of an interface with four-finger pitch selection. In *Proceedings of the 142nd Audio Engineering Society Convention*. Berlin, Germany.
- von Hornbostel, E. & Sachs, C. (1914). Systematik der musikinstrumente. Ein Versuch. *Zeitschrift für Ethnologie*, 46(H. 4/5), 553–590.
- Wager, S., Chen, L., Kim, M., & Raphael, C. (2017). Towards expressive instrument synthesis through smooth frame-by-frame reconstruction: From string to woodwind. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 391–395). New Orleans, USA.
- Wanderley, M. & Depalle, P. (2004). Gestural control of sound synthesis. *Proceedings of the IEEE*, 92(4), 632–644.

REFERENCES

- Warusfel, O. & Misdariis, N. (2001). Directivity synthesis with a 3D array of loudspeakers: Application for stage performance. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx)* (pp. 1–5). Limerick, Ireland.
- Weidenaar, R. (1995). *Magic music from the Telharmonium*. The Scarecrow Press.
- Wessel, D. L. (1979). Timbre space as a musical control structure. *Computer Music Journal*, 3(2), 45–52.
- Wessel, D., Drame, C., & Wright, M. (1998). Removing the time axis from spectral model analysis-based additive synthesis: Neural networks versus memory-based machine learning. In *Proceedings of the International Computer Music Conference (ICMC)* (pp. 62–65). Ann Arbor, Michigan.
- Westerman, K. N. (1938). The physiology of vibrato. *Music Educators Journal*, 24, 48–49.
- Widmer, G. & Goebel, W. (2004). Computational models of expressive music performance: The state of the art. *Journal of New Music Research*, 33, 203–216.
- Wiemann, B. (2017). *Implementation of an additive sound synthesis for an electronic musical instrument* (Master's thesis, TU Berlin).
- Wilcox, R. (2011). *Modern statistics for the social and behavioral sciences: A practical introduction* (2nd ed.). CRC press.
- Wilcox, R. (2020, October 20). WRS software. Retrieved from <https://dornsife.usc.edu/labs/rwilcox/software/>
- Wilson, S. (2008). Spatial swarm granulation. In *Proceedings of the International Computer Music Conference (ICMC)*. Belfast, Ireland.
- Wilson, S. & Harrison, J. (2010). Rethinking the BEAST: Recent developments in multichannel composition at birmingham electroacoustic sound theatre. *Organised Sound*, 15(3), 239–250.
- Yang, L., Rajab, K. Z., & Chew, E. (2017). The filter diagonalisation method for music signal analysis: Frame-wise vibrato detection and estimation. *Journal of Mathematics and Music*, 11(1), 42–60.
- Young, D. (2002). The Hyperbow controller: Real-time dynamics measurement of violin performance. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)* (pp. 1–6). Dublin, Ireland.

REFERENCES

- Zacharov, N. & Koivuniemi, K. (2001). Unravelling the perception of spatial sound reproduction: Language development, verbal protocol analysis and listener training. In *Proceedings of the 111th Audio Engineering Society Convention*. New York, NY, USA.
- Zen, H., Tokuda, K., & Black, A. W. (2009). Statistical parametric speech synthesis. *Speech Communication*, 51(11), 1039–1064
- Zwicker, E. (1961). Subdivision of the audible frequency range into critical bands (Frequenzgruppen). *The Journal of the Acoustical Society of America*, 33(2), 248–248.

Index

A

Aftertouch	61
Azimuth	67

B

Blurring	51
----------------	----

C

CMF	115
Concatenative Synthesis	33

D

Data Gloves	63
Detached	20, 22
Deterministic Component	35
Deterministic Mode	133
Deterministic Model	130
Discrete Dispersion	140
Distance	68

E

Elektronische Musik	66
Elevation	67
Expressive Capabilities	42
Expressive Musical Content	17
Expressiveness	17
Expressivity	17

F

Filter Bank Dispersion	140
FSR	165
Fusion-Mapping-Fission	74

G

Granular Synthesis	32
--------------------------	----

H

Harmonic Spectral Centroid	163
HSC	163

I

ICMF	115
Intra-Note-Segmentation	95
Inverse Transform Sampling	136
Inverse Transform Sampling Synthesis	135

J

Jitter	108
--------------	-----

K

Key Overlap Time	21
------------------------	----

L

Legato	20
Legato Index	22

M

Mapping	73
Markovian Statistical Modeling	118
Mean Mode	135
MITSSy	137
Mod Wheel	61
Modwheel	65

N

Note-Transition-Rest	95
----------------------------	----

O

Object-Based Spatialization	50
Ondes Martenot	60
Open Sound Control	143

OSC	143
Overshoots	22

P

Partial Performance Synthesis	65
Partial Tracking	35
Performance Model	65
Performance Rendering	64
Performance Synthesis	64
Pitch Wheel	61
PMF	114
Point-Source Panning	50
Preparations	22
Pseudo Polyphony	101, 148

Q

Quantile Function	115
-------------------------	-----

R

Reconstructive Phrase Modeling	42
Residual	35
RPM	42

S

Sample Library	84
Sampling	32
Scattering	51
Shimmer	108
SIM	166
Sinusoidal Modeling	34
Sinusoidal Modeling Synthesis	35
Slope Overload Memory	109
SMS	35
SMS Rate	149
Sound Diffusion	47

Sound Reproduction System	47
Spatial Additive Synthesis	54
Spatial Sound Synthesis	49
Spatialization	47
Spatiomorphology	47
Spectral Delay	48
Spectral Dispersion	139
Spectral Spatialization	48
Spectro-Spatial Dispersion	181
Spectro-Spatial Sound Synthesis	53
Stateless Inverse Transform Sampling Synthesis 135	
Stateless Statistical Modeling	114
Statistical Model	130
Strictly Harmonic Model	36

T

Theremin	60
Timbre Plane	86
Timbre Spatialization	48
Timbre-Space Interpolation	133
Topographic Synthesis	53
Transition Probabilities	118
Trautonium	60

U

UEQ	194
Uniform Dispersion	139

V

Valve Mechanism	165
-----------------------	-----

W

Wave Terrain Synthesis	48
Williams Mix	32

List of Figures

1.1	Relevant components of the project and their relation.	14
2.1	Three modes of inter-note articulation after Strawn (1985).	21
2.2	Durational parameters of notes and transitions after Bresin (2001).	21
2.3	RMS trajectory of two-note item 123 from the sample library with areas for calculating the legato index.	22
2.4	Fundamental frequency trajectory of a glissando note transition.	23
2.5	Spectral envelope modulations for the ninth partial of an excerpt from two-note item 22.	26
2.6	Local minima and maxima within an idealized modulation trajectory.	27
2.7	Frequency domain analysis of a vibrato modulation trajectory.	28
3.1	Examples for Blackman-Harris main-lobe kernels at different shifts between two neighboring frequency bins.	40
3.2	Two adjacent sinusoidal frames with a triangular window and phase matching.	41
4.1	Taxonomy of muscle-based interaction, proposed by Bongers (2000).	57
4.2	General digital musical instrument model, featuring interface, mapping and sound synthesis.	73
4.3	Mapping sensors to synthesis, using the <i>Fusion-Mapping-Fission</i> model.	74
5.1	GLOOO project overview with the four components developed in this thesis and the third party rendering software.	83
6.1	The timbre plane with a visualization of the support points.	87
6.2	Violin fingerboard with intervals captured in the two-note section.	88
6.3	Position of the DPA microphone on the violin.	91

LIST OF FIGURES

6.4	Michiko Feuerlein performing in the anechoic chamber with the B&K microphone in the upper right.	92
6.5	Micro-scores from the recording instructions.	94
6.6	End of the attack segment through RMS threshold method and attack segment obtained through spectral criterion.	96
6.7	Beginning of the release segment through RMS threshold method and release segment obtained through spectral criterion.	96
6.8	Segmentation setup in SonicVisualiser for two-note item 21 with vibrato and glissando.	97
6.9	Segmentation of single note item 333 with SonicVisualiser the time domain.	97
6.10	Segmentation of single item 333 with SonicVisualiser in the time-frequency domain.	98
7.1	Block chart of the analysis and modeling process.	100
7.2	Estimating the phase φ_i of a partial by subtraction.	103
7.3	Amplitude trajectories of the first 20 partials for item 22 from the single sounds of the sample library.	104
7.4	Frequency trajectories of the first 20 partials for item 22 from the single sounds of the sample library.	105
7.5	Unwrapped partial phases of the first 20 partials for item 22 from the single sounds of the sample library.	105
7.6	Amplitude responses of the individual IIR filters in the Bark frequency scale filter bank.	106
7.7	Energy trajectory of the first Bark band for item 08 from the single sounds of the sample library.	106
7.8	Energy trajectory of the second Bark band for item 08 from the single sounds of the sample library.	107
7.9	Energy trajectory of the fourth Bark band for item 08 from the single sounds of the sample library.	107
7.10	Energy trajectory of the eighth Bark band for item 08 from the single sounds of the sample library.	107
7.11	Energy trajectory of the 16th Bark band for item 08 from the single sounds of the sample library.	107
7.12	Flow chart of the SOM - f_0 decomposition algorithm (von Coler & Röbel, 2011).	110
7.13	Normalized frequency trajectory for the first partial in the sustain part of single sound item 40.	111
7.14	Probability density functions of partial frequencies during the sustain of single sound item 40.	111
7.15	Decomposition of the first partial's amplitude trajectory for the sustain part of single sound item 40.	112
7.16	Probability density functions of partial amplitudes with separate plots for the complete trajectories (gray) and after decomposition (black).	112

LIST OF FIGURES

7.17	Decomposition of the residual energy trajectory for Bark band six in the sustain part of single sound item 40.	113
7.18	Probability density functions of Bark band energies for the original trajectory (gray) and after pre-processing (black).	113
7.19	Example of a probability mass function (PMF).	114
7.20	Example of a cumulative mass function (CMF).	115
7.21	Example of an inverse cumulative mass function (ICMF).	116
7.22	CMFs for the first 30 partial amplitudes of single sound item 55.	117
7.23	CMFs for the first 30 partial frequencies of single sound item 55.	117
7.24	CMFs for the 24 bark energy trajectories of single sound item 55.	117
7.25	Probability mass function of the pre-processed amplitude trajectory for the second partial of single sound item 01 with 21 quantization stages.	118
7.26	Individual transition probabilities $P_i(a)$ for all 21 quantization stages i of the first partial amplitude trajectory.	119
7.27	Individual transition probabilities $P_i(f)$ for all quantization stages i of the sixth partial frequency trajectory in single sound item 01.	120
7.28	Individual transition probabilities $P_i(f)$ for all quantization stages of the fifth Bark band energy in single sound item 01.	120
7.29	Inverted cumulative mass functions for the transition probabilities of the first partial's amplitude trajectory in single sound item 01.	121
7.30	Inverted cumulative mass functions for the transition probabilities of the first partial's frequency trajectory in single sound item 01.	122
7.31	Inverted cumulative mass functions for the transition probabilities of the fifth Bark band energy trajectory in single sound item 01.	123
7.32	Release trajectories from the exponential model with different values for λ .	125
7.33	Normalized original and modeled release trajectories of the first four partials from item 40 of the single sound items.	125
7.34	Mean λ values (gray) and smoothed version (black) for all partials, grouped according to Table 7.1.	127
7.35	Hyperbolic tangent of the length N with different values for λ (von Coler, Götz, & Lepa, 2018).	128
7.36	Corrected hyperbolic tangent with different values for b .	129
7.37	YAML data structure for a single item in the stochastic data set.	131
8.1	Support points A , B , C and D with an arbitrary point P in the timbre plane.	134
8.2	Interpolated inverse transform sampling.	137

LIST OF FIGURES

8.3	Flow chart for the Markovian Inverse Transform Sampling Synthesis for a single parameter with the algorithm for interpolated inverse transform sampling from Figure 8.2 inside the gray box.	138
8.4	Uniform mapping of 80 partials and 24 noise bands to M sound sources.	139
8.5	The discrete dispersion mode with 80 partials, 24 noise band signals and 104 point sound sources.	140
8.6	Filter bank dispersion with 24 Bark scale filters.	141
8.7	Gaussian approximations of the Bark band filter bank.	142
8.8	GLOOO synthesizer class diagram.	147
8.9	State diagram for the <code>single_voice</code> class.	148
8.10	Number of synthesized partials for different fundamental frequencies and sampling rates.	150
8.11	Sequence diagram for the JACK callback process in the filter bank dispersion mode.	151
8.12	Input parameters of the GLOOO synthesis program.	154
8.13	Amplitude over frequency for the first partial during a pitch sweep of one octave.	156
8.14	Amplitude over frequency for the second partial during a pitch sweep of one octave.	157
8.15	Amplitude over frequency for the third partial during a pitch sweep of one octave.	157
8.16	Approximation of the frequency response through a pitch sweep.	158
8.17	Input admittance at the bass bar side of the Andrea Guarneri violin (Alonso Moral & Jansson, 1982).	158
8.18	Approximations of the frequency response through spectral envelope modulations of 30 partials by a one octave sweep.	160
8.19	Harmonic spectral envelopes for four different pitches, sampled at 10 instances during intensity sweeps.	162
8.20	Harmonic Spectral Centroid as function of intensity at four different pitches.	163
9.1	First design study for a tabletop device with four FSRs and three octave switches.	165
9.2	First hand-held design without excitation.	166
9.3	Design studies I by Gabriel Treindl (von Coler, Treindl, Egermann, & Weinzierl, 2017).	166
9.4	Design studies II by Gabriel Treindl (von Coler, Treindl, Egermann, & Weinzierl, 2017).	167
9.5	Valve mechanism.	167
9.6	Front (a) and rear (b) view of the BINBONG MKI.	168
9.7	The BINBONG MKII.	169
9.8	Longitudinal section of the BINBONG, side view (Schmied, 2018).	170
9.9	Wooden top with excitation pad, four FSRs and four felt inserts (Schmied, 2018).	170
9.10	Force-based sensor units.	171
9.11	Four valves.	172

LIST OF FIGURES

9.12	Force value of the pad.	174
9.13	Position value of the pad.	174
9.14	Position value from the ribbon sensor.	175
9.15	Force value from the ribbon sensor.	175
9.16	Swipe gesture of the ribbon controller.	175
9.17	Three rotation axes of the BinBong.	176
9.18	Rotation around the x-axis.	176
9.19	Rotation around the y-axis.	177
9.20	Rotation around the z-axis.	177
9.21	Control parameters sent from the BinBong.	178
9.22	The extended DMI model for spatial sound synthesis.	180
9.23	BinBong receiver patch.	181
9.24	Virtual sound source with a position and spatial extent.	182
9.25	Patch for sending OSC to the spatial rendering software.	182
9.26	GLOOO parameter settings patch.	183
9.27	Patch for sending OSC to the synthesis software.	184
9.28	Mapping GUI as used in the user study, fully patched.	184
10.1	Test setup in the small studio (EN 325) at TU Berlin.	188
10.2	Geometry of the dome with 21 loudspeakers in the TU Studio (EN 325).	189
10.3	Control interface for the gesture stage.	193
10.4	7 point Likert scale for the gesture survey.	194
10.5	Average experience of participants from self-assessment.	195
10.6	GMSI results.	196
10.7	General Sophistication index for the individual participants.	197
10.8	UEQ benchmark results.	198
10.9	Box plots with ratings for each attribute.	198
10.10	Box plots with ratings from each participant for all 12 attributes of the survey.	199
10.11	Matrix with the number of individual connections made to the rendering parameters for each participant.	201
10.12	Number of individual control parameters connected to the rendering parameters, grouped by participant.	201
10.13	Normalized frequency of individual connections for the rendering categories.	202
10.14	Matrix with the number of individual connections made to the control parameters by each participant.	203

LIST OF FIGURES

10.15	Number of individual rendering parameters connected to the control parameters, grouped by participant.	204
10.16	Normalized frequency of individual connections for the control units.	204
10.17	Mapping frequency for the control parameters over all participants in the final mappings.	205
10.18	Mapping matrix, showing the total number of connections over all participants.	206
10.19	Matrix with mapping similarities between participants.	207
10.20	Mapping process of participant 08.	211
10.21	Mapping process of participant 14.	212
10.22	Mapping process of participant 05.	212

List of Tables

3.1	Taxonomy of synthesis approaches proposed by J. O. Smith (1991).	30
3.2	Bark scale frequency bands with center and cutoff frequencies.	38
4.1	Occurrences of different sensor types in NIME instruments (M. T. Marshall, Hartshorn, Wanderley, & Levitin, 2009).	57
4.2	Perceptual qualities in the category <i>geometry</i> of the SAQI (Lindau, 2015).	68
4.3	Composers' responses on spatial compositional means (Peters, 2011, p.34).	69
6.1	Combination matrix for the different expressive styles to be recorded.	89
6.2	Library items as combinations of the two scales with three expressive levels.	89
6.3	Compositions included in the solo part of the sample library (von Coler, Margraf, & Schuladen, 2018).	90
6.4	Expressivity marks used in the score snippets.	93
6.5	String numbers used in the instructions.	93
7.1	Subsets for the release modeling	126
8.1	Parameters of the Gaussian filter approximations.	142
8.2	Libraries installed on the development system	143
8.3	Assignment file and filter model in the configuration file.	152
8.4	Basic synthesis settings the configuration file.	152
8.5	Communication parameters in the configuration file.	153
8.6	Debug output parameters in the configuration file.	153
8.7	Performance Parameters.	155
8.8	Tuning Parameters.	155

LIST OF TABLES

8.9	Synthesis Modes.	155
8.10	Partial Activity.	155
8.11	Main resonances of a violin body, revisited by Curtin and Rossing (2010).	159
8.12	Main resonances from the pitch sweep measurements in Figure 8.16.	159
9.1	Binary Code Mapping - <i>F5 = little finger, F4 = ring finger, F3 = middle finger, F2 = index</i> (von Coler, Treindl, Egermann, & Weinzierl, 2017).	173
9.2	OSC messages for the valves.	178
9.3	OSC messages for the pad.	179
9.4	OSC messages for ribbon controller.	179
9.5	OSC messages for acceleration and orientation.	179
9.6	OSC messages for IMU calibration.	179
10.1	Control parameters and rendering parameters in the user study.	190
10.2	The four mapping directives.	191
10.3	Questions in the mapping survey.	191
10.4	Single instructions for the gesture stage.	192
10.5	12 items in the gesture survey.	193
10.6	Experience-related statements in the survey.	194
10.7	General feedback on the full system.	194
10.8	Statistics of 147,633 participants from the GMSI (Müllensiefen, Gingras, Musil, & Stewart, 2014).	196
10.9	Mean values and variances for the major scales of the UEQ.	197
10.10	Three main qualities of the system in the UEQ.	197
10.11	Two groups of different average satisfaction.	199
10.12	Time used for the mapping process and total number of mapping changes for each partici- pant.	200
10.13	Grouping of rendering parameters into three categories for the analysis of the exploration activity.	202
10.14	Grouping of control parameters by unit for the analysis of the exploration activity.	203
10.15	Chi ² tests for all control parameters	205
10.16	Mean similarity score for each participant.	206

Appendix I

Single Sounds List

File	String	Pos	MIDI	ISO	Dyn	Octave
SampLib_DPA_01	G	0	55	G3	pp	1
SampLib_DPA_02	G	0	55	G3	mp	1
SampLib_DPA_03	G	0	55	G3	mf	1
SampLib_DPA_04	G	0	55	G3	ff	1
SampLib_DPA_05	G	1	56	G#3	pp	1
SampLib_DPA_06	G	1	56	G#3	mp	1
SampLib_DPA_07	G	1	56	G#3	mf	1
SampLib_DPA_08	G	1	56	G#3	ff	1
SampLib_DPA_09	G	2	57	A3	pp	1
SampLib_DPA_10	G	2	57	A3	mp	1
SampLib_DPA_11	G	2	57	A3	mf	1
SampLib_DPA_12	G	2	57	A3	ff	1
SampLib_DPA_13	G	3	58	A#3	pp	1
SampLib_DPA_14	G	3	58	A#3	mp	1
SampLib_DPA_15	G	3	58	A#3	mf	1
SampLib_DPA_16	G	3	58	A#3	ff	1
SampLib_DPA_17	G	4	59	B3	pp	2
SampLib_DPA_18	G	4	59	B3	mp	2
SampLib_DPA_19	G	4	59	B3	mf	2
SampLib_DPA_20	G	4	59	B3	ff	2
SampLib_DPA_21	G	5	60	C4	pp	2
SampLib_DPA_22	G	5	60	C4	mp	2
SampLib_DPA_23	G	5	60	C4	mf	2
SampLib_DPA_24	G	5	60	C4	ff	2
SampLib_DPA_25	G	6	61	C#4	pp	2
SampLib_DPA_26	G	6	61	C#4	mp	2
SampLib_DPA_27	G	6	61	C#4	mf	2
SampLib_DPA_28	G	6	61	C#4	ff	2
SampLib_DPA_29	G	7	62	D4	pp	2

APPENDIX I. SINGLE SOUNDS LIST

SampLib_DPA_30	G	7	62	D4	mp	2
SampLib_DPA_31	G	7	62	D4	mf	2
SampLib_DPA_32	G	7	62	D4	ff	2
SampLib_DPA_33	G	8	63	D#4	pp	2
SampLib_DPA_34	G	8	63	D#4	mp	2
SampLib_DPA_35	G	8	63	D#4	mf	2
SampLib_DPA_36	G	8	63	D#4	ff	2
SampLib_DPA_37	G	9	64	E4	pp	2
SampLib_DPA_38	G	9	64	E4	mp	2
SampLib_DPA_39	G	9	64	E4	mf	2
SampLib_DPA_40	G	9	64	E4	ff	2
SampLib_DPA_41	G	10	65	F4	pp	2
SampLib_DPA_42	G	10	65	F4	mp	2
SampLib_DPA_43	G	10	65	F4	mf	2
SampLib_DPA_44	G	10	65	F4	ff	2
SampLib_DPA_45	G	11	66	F#4	pp	2
SampLib_DPA_46	G	11	66	F#4	mp	2
SampLib_DPA_47	G	11	66	F#4	mf	2
SampLib_DPA_48	G	11	66	F#4	ff	2
SampLib_DPA_49	G	12	67	G4	pp	2
SampLib_DPA_50	G	12	67	G4	mp	2
SampLib_DPA_51	G	12	67	G4	mf	2
SampLib_DPA_52	G	12	67	G4	ff	2
SampLib_DPA_53	G	13	68	G#4	pp	2
SampLib_DPA_54	G	13	68	G#4	mp	2
SampLib_DPA_55	G	13	68	G#4	mf	2
SampLib_DPA_56	G	13	68	G#4	ff	2
SampLib_DPA_57	G	14	69	A4	pp	2
SampLib_DPA_58	G	14	69	A4	mp	2
SampLib_DPA_59	G	14	69	A4	mf	2
SampLib_DPA_60	G	14	69	A4	ff	2
SampLib_DPA_61	G	15	70	A#4	pp	3
SampLib_DPA_62	G	15	70	A#4	mp	3
SampLib_DPA_63	G	15	70	A#4	mf	3
SampLib_DPA_64	G	15	70	A#4	ff	3
SampLib_DPA_65	G	16	71	B4	pp	3
SampLib_DPA_66	G	16	71	B4	mp	3
SampLib_DPA_67	G	16	71	B4	mf	3
SampLib_DPA_68	G	16	71	B4	ff	3
SampLib_DPA_69	G	17	72	C5	pp	3
SampLib_DPA_70	G	17	72	C5	mp	3
SampLib_DPA_71	G	17	72	C5	mf	3
SampLib_DPA_72	G	17	72	C5	ff	3
SampLib_DPA_73	D	0	62	D4	pp	2
SampLib_DPA_74	D	0	62	D4	mp	2
SampLib_DPA_75	D	0	62	D4	mf	2

APPENDIX I. SINGLE SOUNDS LIST

SampLib_DPA_76	D	0	62	D4	ff	2
SampLib_DPA_77	D	1	63	D#4	pp	2
SampLib_DPA_78	D	1	63	D#4	mp	2
SampLib_DPA_79	D	1	63	D#4	mf	2
SampLib_DPA_80	D	1	63	D#4	ff	2
SampLib_DPA_81	D	2	64	E4	pp	2
SampLib_DPA_82	D	2	64	E4	mp	2
SampLib_DPA_83	D	2	64	E4	mf	2
SampLib_DPA_84	D	2	64	E4	ff	2
SampLib_DPA_85	D	3	65	F4	pp	3
SampLib_DPA_86	D	3	65	F4	mp	3
SampLib_DPA_87	D	3	65	F4	mf	3
SampLib_DPA_88	D	3	65	F4	ff	3
SampLib_DPA_89	D	4	66	F#4	pp	3
SampLib_DPA_90	D	4	66	F#4	mp	3
SampLib_DPA_91	D	4	66	F#4	mf	3
SampLib_DPA_92	D	4	66	F#4	ff	3
SampLib_DPA_93	D	5	67	G4	pp	3
SampLib_DPA_94	D	5	67	G4	mp	3
SampLib_DPA_95	D	5	67	G4	mf	3
SampLib_DPA_96	D	5	67	G4	ff	3
SampLib_DPA_97	D	6	68	G#4	pp	3
SampLib_DPA_98	D	6	68	G#4	mp	3
SampLib_DPA_99	D	6	68	G#4	mf	3
SampLib_DPA_100	D	6	68	G#4	ff	3
SampLib_DPA_101	D	7	69	A4	pp	3
SampLib_DPA_102	D	7	69	A4	mp	3
SampLib_DPA_103	D	7	69	A4	mf	3
SampLib_DPA_104	D	7	69	A4	ff	3
SampLib_DPA_105	D	8	70	A#4	pp	3
SampLib_DPA_106	D	8	70	A#4	mp	3
SampLib_DPA_107	D	8	70	A#4	mf	3
SampLib_DPA_108	D	8	70	A#4	ff	3
SampLib_DPA_109	D	9	71	B4	pp	3
SampLib_DPA_110	D	9	71	B4	mp	3
SampLib_DPA_111	D	9	71	B4	mf	3
SampLib_DPA_112	D	9	71	B4	ff	3
SampLib_DPA_113	D	10	72	C5	pp	3
SampLib_DPA_114	D	10	72	C5	mp	3
SampLib_DPA_115	D	10	72	C5	mf	3
SampLib_DPA_116	D	10	72	C5	ff	3
SampLib_DPA_117	D	11	73	C#5	pp	3
SampLib_DPA_118	D	11	73	C#5	mp	3
SampLib_DPA_119	D	11	73	C#5	mf	3
SampLib_DPA_120	D	11	73	C#5	ff	3
SampLib_DPA_121	D	12	74	D5	pp	3

APPENDIX I. SINGLE SOUNDS LIST

SampLib_DPA_122	D	12	74	D5	mp	3
SampLib_DPA_123	D	12	74	D5	mf	3
SampLib_DPA_124	D	12	74	D5	ff	3
SampLib_DPA_125	D	13	75	D#5	pp	3
SampLib_DPA_126	D	13	75	D#5	mp	3
SampLib_DPA_127	D	13	75	D#5	mf	3
SampLib_DPA_128	D	13	75	D#5	ff	3
SampLib_DPA_129	D	14	76	E5	pp	4
SampLib_DPA_130	D	14	76	E5	mp	4
SampLib_DPA_131	D	14	76	E5	mf	4
SampLib_DPA_132	D	14	76	E5	ff	4
SampLib_DPA_133	D	15	77	F5	pp	4
SampLib_DPA_134	D	15	77	F5	mp	4
SampLib_DPA_135	D	15	77	F5	mf	4
SampLib_DPA_136	D	15	77	F5	ff	4
SampLib_DPA_137	D	16	78	F#5	pp	4
SampLib_DPA_138	D	16	78	F#5	mp	4
SampLib_DPA_139	D	16	78	F#5	mf	4
SampLib_DPA_140	D	16	78	F#5	ff	4
SampLib_DPA_141	D	17	79	G5	pp	4
SampLib_DPA_142	D	17	79	G5	mp	4
SampLib_DPA_143	D	17	79	G5	mf	4
SampLib_DPA_144	D	17	79	G5	ff	4
SampLib_DPA_145	A	0	69	A4	pp	3
SampLib_DPA_146	A	0	69	A4	mp	3
SampLib_DPA_147	A	0	69	A4	mf	3
SampLib_DPA_148	A	0	69	A4	ff	3
SampLib_DPA_149	A	1	70	A#4	pp	3
SampLib_DPA_150	A	1	70	A#4	mp	3
SampLib_DPA_151	A	1	70	A#4	mf	3
SampLib_DPA_152	A	1	70	A#4	ff	3
SampLib_DPA_153	A	2	71	B4	pp	4
SampLib_DPA_154	A	2	71	B4	mp	4
SampLib_DPA_155	A	2	71	B4	mf	4
SampLib_DPA_156	A	2	71	B4	ff	4
SampLib_DPA_157	A	3	72	C5	pp	4
SampLib_DPA_158	A	3	72	C5	mp	4
SampLib_DPA_159	A	3	72	C5	mf	4
SampLib_DPA_160	A	3	72	C5	ff	4
SampLib_DPA_161	A	4	73	C#5	pp	4
SampLib_DPA_162	A	4	73	C#5	mp	4
SampLib_DPA_163	A	4	73	C#5	mf	4
SampLib_DPA_164	A	4	73	C#5	ff	4
SampLib_DPA_165	A	5	74	D5	pp	4
SampLib_DPA_166	A	5	74	D5	mp	4
SampLib_DPA_167	A	5	74	D5	mf	4

APPENDIX I. SINGLE SOUNDS LIST

SampLib_DPA_168	A	5	74	D5	ff	4
SampLib_DPA_169	A	6	75	D#5	pp	4
SampLib_DPA_170	A	6	75	D#5	mp	4
SampLib_DPA_171	A	6	75	D#5	mf	4
SampLib_DPA_172	A	6	75	D#5	ff	4
SampLib_DPA_173	A	7	76	E5	pp	4
SampLib_DPA_174	A	7	76	E5	mp	4
SampLib_DPA_175	A	7	76	E5	mf	4
SampLib_DPA_176	A	7	76	E5	ff	4
SampLib_DPA_177	A	8	77	F5	pp	4
SampLib_DPA_178	A	8	77	F5	mp	4
SampLib_DPA_179	A	8	77	F5	mf	4
SampLib_DPA_180	A	8	77	F5	ff	4
SampLib_DPA_181	A	9	78	F#5	pp	4
SampLib_DPA_182	A	9	78	F#5	mp	4
SampLib_DPA_183	A	9	78	F#5	mf	4
SampLib_DPA_184	A	9	78	F#5	ff	4
SampLib_DPA_185	A	10	79	G5	pp	4
SampLib_DPA_186	A	10	79	G5	mp	4
SampLib_DPA_187	A	10	79	G5	mf	4
SampLib_DPA_188	A	10	79	G5	ff	4
SampLib_DPA_189	A	11	80	G#5	pp	4
SampLib_DPA_190	A	11	80	G#5	mp	4
SampLib_DPA_191	A	11	80	G#5	mf	4
SampLib_DPA_192	A	11	80	G#5	ff	4
SampLib_DPA_193	A	12	81	A5	pp	4
SampLib_DPA_194	A	12	81	A5	mp	4
SampLib_DPA_195	A	12	81	A5	mf	4
SampLib_DPA_196	A	12	81	A5	ff	4
SampLib_DPA_197	A	13	82	A#5	pp	5
SampLib_DPA_198	A	13	82	A#5	mp	5
SampLib_DPA_199	A	13	82	A#5	mf	5
SampLib_DPA_200	A	13	82	A#5	ff	5
SampLib_DPA_201	A	14	83	B5	pp	5
SampLib_DPA_202	A	14	83	B5	mp	5
SampLib_DPA_203	A	14	83	B5	mf	5
SampLib_DPA_204	A	14	83	B5	ff	5
SampLib_DPA_205	A	15	84	C6	pp	5
SampLib_DPA_206	A	15	84	C6	mp	5
SampLib_DPA_207	A	15	84	C6	mf	5
SampLib_DPA_208	A	15	84	C6	ff	5
SampLib_DPA_209	A	16	85	C#6	pp	5
SampLib_DPA_210	A	16	85	C#6	mp	5
SampLib_DPA_211	A	16	85	C#6	mf	5
SampLib_DPA_212	A	16	85	C#6	ff	5
SampLib_DPA_213	A	17	86	D6	pp	5

APPENDIX I. SINGLE SOUNDS LIST

SampLib_DPA_214	A	17	86	D6	mp	5
SampLib_DPA_215	A	17	86	D6	mf	5
SampLib_DPA_216	A	17	86	D6	ff	5
SampLib_DPA_217	E	0	76	E5	pp	4
SampLib_DPA_218	E	0	76	E5	mp	4
SampLib_DPA_219	E	0	76	E5	mf	4
SampLib_DPA_220	E	0	76	E5	ff	4
SampLib_DPA_221	E	1	77	F5	pp	5
SampLib_DPA_222	E	1	77	F5	mp	5
SampLib_DPA_223	E	1	77	F5	mf	5
SampLib_DPA_224	E	1	77	F5	ff	5
SampLib_DPA_225	E	2	78	F#5	pp	5
SampLib_DPA_226	E	2	78	F#5	mp	5
SampLib_DPA_227	E	2	78	F#5	mf	5
SampLib_DPA_228	E	2	78	F#5	ff	5
SampLib_DPA_229	E	3	79	G5	pp	5
SampLib_DPA_230	E	3	79	G5	mp	5
SampLib_DPA_231	E	3	79	G5	mf	5
SampLib_DPA_232	E	3	79	G5	ff	5
SampLib_DPA_233	E	4	80	G#5	pp	5
SampLib_DPA_234	E	4	80	G#5	mp	5
SampLib_DPA_235	E	4	80	G#5	mf	5
SampLib_DPA_236	E	4	80	G#5	ff	5
SampLib_DPA_237	E	5	81	A5	pp	5
SampLib_DPA_238	E	5	81	A5	mp	5
SampLib_DPA_239	E	5	81	A5	mf	5
SampLib_DPA_240	E	5	81	A5	ff	5
SampLib_DPA_241	E	6	82	A#5	pp	5
SampLib_DPA_242	E	6	82	A#5	mp	5
SampLib_DPA_243	E	6	82	A#5	mf	5
SampLib_DPA_244	E	6	82	A#5	ff	5
SampLib_DPA_245	E	7	83	B5	pp	5
SampLib_DPA_246	E	7	83	B5	mp	5
SampLib_DPA_247	E	7	83	B5	mf	5
SampLib_DPA_248	E	7	83	B5	ff	5
SampLib_DPA_249	E	8	84	C6	pp	5
SampLib_DPA_250	E	8	84	C6	mp	5
SampLib_DPA_251	E	8	84	C6	mf	5
SampLib_DPA_252	E	8	84	C6	ff	5
SampLib_DPA_253	E	9	85	C#6	pp	5
SampLib_DPA_254	E	9	85	C#6	mp	5
SampLib_DPA_255	E	9	85	C#6	mf	5
SampLib_DPA_256	E	9	85	C#6	ff	5
SampLib_DPA_257	E	10	86	D6	pp	5
SampLib_DPA_258	E	10	86	D6	mp	5
SampLib_DPA_259	E	10	86	D6	mf	5

APPENDIX I. SINGLE SOUNDS LIST

SampLib_DPA_260	E	10	86	D6	ff	5
SampLib_DPA_261	E	11	87	D#6	pp	5
SampLib_DPA_262	E	11	87	D#6	mp	5
SampLib_DPA_263	E	11	87	D#6	mf	5
SampLib_DPA_264	E	11	87	D#6	ff	5
SampLib_DPA_265	E	12	88	E6	pp	6
SampLib_DPA_266	E	12	88	E6	mp	6
SampLib_DPA_267	E	12	88	E6	mf	6
SampLib_DPA_268	E	12	88	E6	ff	6
SampLib_DPA_269	E	13	89	F6	pp	6
SampLib_DPA_270	E	13	89	F6	mp	6
SampLib_DPA_271	E	13	89	F6	mf	6
SampLib_DPA_272	E	13	89	F6	ff	6
SampLib_DPA_273	E	14	90	F#6	pp	6
SampLib_DPA_274	E	14	90	F#6	mp	6
SampLib_DPA_275	E	14	90	F#6	mf	6
SampLib_DPA_276	E	14	90	F#6	ff	6
SampLib_DPA_277	E	15	91	G6	pp	6
SampLib_DPA_278	E	15	91	G6	mp	6
SampLib_DPA_279	E	15	91	G6	mf	6
SampLib_DPA_280	E	15	91	G6	ff	6
SampLib_DPA_281	E	16	92	G#6	pp	6
SampLib_DPA_282	E	16	92	G#6	mp	6
SampLib_DPA_283	E	16	92	G#6	mf	6
SampLib_DPA_284	E	16	92	G#6	ff	6
SampLib_DPA_285	E	17	93	A6	pp	6
SampLib_DPA_286	E	17	93	A6	mp	6
SampLib_DPA_287	E	17	93	A6	mf	6
SampLib_DPA_288	E	17	93	A6	ff	6
SampLib_DPA_289	E	18	94	A#6	pp	6
SampLib_DPA_290	E	18	94	A#6	mp	6
SampLib_DPA_291	E	18	94	A#6	mf	6
SampLib_DPA_292	E	18	94	A#6	ff	6
SampLib_DPA_293	E	19	95	B6	pp	6
SampLib_DPA_294	E	19	95	B6	mp	6
SampLib_DPA_295	E	19	95	B6	mf	6
SampLib_DPA_296	E	19	95	B6	ff	6
SampLib_DPA_297	E	20	96	C7	pp	6
SampLib_DPA_298	E	20	96	C7	mp	6
SampLib_DPA_299	E	20	96	C7	mf	6
SampLib_DPA_300	E	20	96	C7	ff	6
SampLib_DPA_301	E	21	97	C#7	pp	6
SampLib_DPA_302	E	21	97	C#7	mp	6
SampLib_DPA_303	E	21	97	C#7	mf	6
SampLib_DPA_304	E	21	97	C#7	ff	6
SampLib_DPA_305	E	22	98	D7	pp	6

APPENDIX I. SINGLE SOUNDS LIST

SampLib_DPA_306	E	22	98	D7	mp	6
SampLib_DPA_307	E	22	98	D7	mf	6
SampLib_DPA_308	E	22	98	D7	ff	6
SampLib_DPA_309	E	23	99	D#7	pp	7
SampLib_DPA_310	E	23	99	D#7	mp	7
SampLib_DPA_311	E	23	99	D#7	mf	7
SampLib_DPA_312	E	23	99	D#7	ff	7
SampLib_DPA_313	E	24	100	E7	pp	7
SampLib_DPA_314	E	24	100	E7	mp	7
SampLib_DPA_315	E	24	100	E7	mf	7
SampLib_DPA_316	E	24	100	E7	ff	7
SampLib_DPA_317	E	25	101	F7	pp	7
SampLib_DPA_318	E	25	101	F7	mp	7
SampLib_DPA_319	E	25	101	F7	mf	7
SampLib_DPA_320	E	25	101	F7	ff	7
SampLib_DPA_321	E	26	102	F#7	pp	7
SampLib_DPA_322	E	26	102	F#7	mp	7
SampLib_DPA_323	E	26	102	F#7	mf	7
SampLib_DPA_324	E	26	102	F#7	ff	7
SampLib_DPA_325	E	27	103	G7	pp	7
SampLib_DPA_326	E	27	103	G7	mp	7
SampLib_DPA_327	E	27	103	G7	mf	7
SampLib_DPA_328	E	27	103	G7	ff	7
SampLib_DPA_329	E	28	104	G#7	pp	7
SampLib_DPA_330	E	28	104	G#7	mp	7
SampLib_DPA_331	E	28	104	G#7	mf	7
SampLib_DPA_332	E	28	104	G#7	ff	7
SampLib_DPA_333	E	29	105	A7	pp	7
SampLib_DPA_334	E	29	105	A7	mp	7
SampLib_DPA_335	E	29	105	A7	mf	7
SampLib_DPA_336	E	29	105	A7	ff	7

Appendix II

Two-Note List

ID	N1	POS1	STR1	N2	POS2	STR2	SEMI	DIR	DYN	ART	VIB
1	D4	7	G	A3	2	G	5	down	mp	detached	false
2	A3	7	G	D4	2	G	5	up	mp	detached	false
3	D4	7	G	A3	2	G	5	down	ff	detached	false
4	A3	7	G	D4	2	G	5	up	ff	detached	false
5	D4	7	G	A3	2	G	5	down	mp	detached	true
6	A3	7	G	D4	2	G	5	up	mp	detached	true
7	D4	7	G	A3	2	G	5	down	ff	detached	true
8	A3	7	G	D4	2	G	5	up	ff	detached	true
9	D4	7	G	A3	2	G	5	down	mp	legato	false
10	A3	7	G	D4	2	G	5	up	mp	legato	false
11	D4	7	G	A3	2	G	5	down	ff	legato	false
12	A3	7	G	D4	2	G	5	up	ff	legato	false
13	D4	7	G	A3	2	G	5	down	mp	legato	true
14	A3	7	G	D4	2	G	5	up	mp	legato	true
15	D4	7	G	A3	2	G	5	down	ff	legato	true
16	A3	7	G	D4	2	G	5	up	ff	legato	true
17	D4	7	G	A3	2	G	5	down	mp	glissando	false
18	A3	7	G	D4	2	G	5	up	mp	glissando	false
19	D4	7	G	A3	2	G	5	down	ff	glissando	false
20	A3	7	G	D4	2	G	5	up	ff	glissando	false
21	D4	7	G	A3	2	G	5	down	mp	glissando	true
22	A3	7	G	D4	2	G	5	up	mp	glissando	true
23	D4	7	G	A3	2	G	5	down	ff	glissando	true
24	A3	7	G	D4	2	G	5	up	ff	glissando	true
25	A4	7	D	E4	2	D	5	down	mp	detached	false
26	E4	7	D	A4	2	D	5	up	mp	detached	false
27	A4	7	D	E4	2	D	5	down	ff	detached	false
28	E4	7	D	A4	2	D	5	up	ff	detached	false
29	A4	7	D	E4	2	D	5	down	mp	detached	true

APPENDIX II. TWO-NOTE LIST

30	E4	7	D	A4	2	D	5	up	mp	detached	true
31	A4	7	D	E4	2	D	5	down	ff	detached	true
32	E4	7	D	A4	2	D	5	up	ff	detached	true
33	A4	7	D	E4	2	D	5	down	mp	legato	false
34	E4	7	D	A4	2	D	5	up	mp	legato	false
35	A4	7	D	E4	2	D	5	down	ff	legato	false
36	E4	7	D	A4	2	D	5	up	ff	legato	false
37	A4	7	D	E4	2	D	5	down	mp	legato	true
38	E4	7	D	A4	2	D	5	up	mp	legato	true
39	A4	7	D	E4	2	D	5	down	ff	legato	true
40	E4	7	D	A4	2	D	5	up	ff	legato	true
41	A4	7	D	E4	2	D	5	down	mp	glissando	false
42	E4	7	D	A4	2	D	5	up	mp	glissando	false
43	A4	7	D	E4	2	D	5	down	ff	glissando	false
44	E4	7	D	A4	2	D	5	up	ff	glissando	false
45	A4	7	D	E4	2	D	5	down	mp	glissando	true
46	E4	7	D	A4	2	D	5	up	mp	glissando	true
47	A4	7	D	E4	2	D	5	down	ff	glissando	true
48	E4	7	D	A4	2	D	5	up	ff	glissando	true
49	E5	7	A	B4	2	A	5	down	mp	detached	false
50	B4	7	A	E5	2	A	5	up	mp	detached	false
51	E5	7	A	B4	2	A	5	down	ff	detached	false
52	B4	7	A	E5	2	A	5	up	ff	detached	false
53	E5	7	A	B4	2	A	5	down	mp	detached	true
54	B4	7	A	E5	2	A	5	up	mp	detached	true
55	E5	7	A	B4	2	A	5	down	ff	detached	true
56	B4	7	A	E5	2	A	5	up	ff	detached	true
57	E5	7	A	B4	2	A	5	down	mp	legato	false
58	B4	7	A	E5	2	A	5	up	mp	legato	false
59	E5	7	A	B4	2	A	5	down	ff	legato	false
60	B4	7	A	E5	2	A	5	up	ff	legato	false
61	E5	7	A	B4	2	A	5	down	mp	legato	true
62	B4	7	A	E5	2	A	5	up	mp	legato	true
63	E5	7	A	B4	2	A	5	down	ff	legato	true
64	B4	7	A	E5	2	A	5	up	ff	legato	true
65	E5	7	A	B4	2	A	5	down	mp	glissando	false
66	B4	7	A	E5	2	A	5	up	mp	glissando	false
67	E5	7	A	B4	2	A	5	down	ff	glissando	false
68	B4	7	A	E5	2	A	5	up	ff	glissando	false
69	E5	7	A	B4	2	A	5	down	mp	glissando	true
70	B4	7	A	E5	2	A	5	up	mp	glissando	true
71	E5	7	A	B4	2	A	5	down	ff	glissando	true
72	B4	7	A	E5	2	A	5	up	ff	glissando	true
73	B5	7	E	F#5	2	E	5	down	mp	detached	false
74	F#5	7	E	B5	2	E	5	up	mp	detached	false
75	B5	7	E	F#5	2	E	5	down	ff	detached	false

APPENDIX II. TWO-NOTE LIST

76	F#5	7	E	B5	2	E	5	up	ff	detached	false
77	B5	7	E	F#5	2	E	5	down	mp	detached	true
78	F#5	7	E	B5	2	E	5	up	mp	detached	true
79	B5	7	E	F#5	2	E	5	down	ff	detached	true
80	F#5	7	E	B5	2	E	5	up	ff	detached	true
81	B5	7	E	F#5	2	E	5	down	mp	legato	false
82	F#5	7	E	B5	2	E	5	up	mp	legato	false
83	B5	7	E	F#5	2	E	5	down	ff	legato	false
84	F#5	7	E	B5	2	E	5	up	ff	legato	false
85	B5	7	E	F#5	2	E	5	down	mp	legato	true
86	F#5	7	E	B5	2	E	5	up	mp	legato	true
87	B5	7	E	F#5	2	E	5	down	ff	legato	true
88	F#5	7	E	B5	2	E	5	up	ff	legato	true
89	B5	7	E	F#5	2	E	5	down	mp	glissando	false
90	F#5	7	E	B5	2	E	5	up	mp	glissando	false
91	B5	7	E	F#5	2	E	5	down	ff	glissando	false
92	F#5	7	E	B5	2	E	5	up	ff	glissando	false
93	B5	7	E	F#5	2	E	5	down	mp	glissando	true
94	F#5	7	E	B5	2	E	5	up	mp	glissando	true
95	B5	7	E	F#5	2	E	5	down	ff	glissando	true
96	F#5	7	E	B5	2	E	5	up	ff	glissando	true
97	D4	7	G	G4	12	G	5	up	mp	detached	false
98	G4	7	G	D4	12	G	5	down	mp	detached	false
99	D4	7	G	G4	12	G	5	up	ff	detached	false
100	G4	7	G	D4	12	G	5	down	ff	detached	false
101	D4	7	G	G4	12	G	5	up	mp	detached	true
102	G4	7	G	D4	12	G	5	down	mp	detached	true
103	D4	7	G	G4	12	G	5	up	ff	detached	true
104	G4	7	G	D4	12	G	5	down	ff	detached	true
105	D4	7	G	G4	12	G	5	up	mp	legato	false
106	G4	7	G	D4	12	G	5	down	mp	legato	false
107	D4	7	G	G4	12	G	5	up	ff	legato	false
108	G4	7	G	D4	12	G	5	down	ff	legato	false
109	D4	7	G	G4	12	G	5	up	mp	legato	true
110	G4	7	G	D4	12	G	5	down	mp	legato	true
111	D4	7	G	G4	12	G	5	up	ff	legato	true
112	G4	7	G	D4	12	G	5	down	ff	legato	true
113	D4	7	G	G4	12	G	5	up	mp	glissando	false
114	G4	7	G	D4	12	G	5	down	mp	glissando	false
115	D4	7	G	G4	12	G	5	up	ff	glissando	false
116	G4	7	G	D4	12	G	5	down	ff	glissando	false
117	D4	7	G	G4	12	G	5	up	mp	glissando	true
118	G4	7	G	D4	12	G	5	down	mp	glissando	true
119	D4	7	G	G4	12	G	5	up	ff	glissando	true
120	G4	7	G	D4	12	G	5	down	ff	glissando	true
121	A4	7	D	D5	12	D	5	up	mp	detached	false

APPENDIX II. TWO-NOTE LIST

122	D5	7	D	A4	12	D	5	down	mp	detached	false
123	A4	7	D	D5	12	D	5	up	ff	detached	false
124	D5	7	D	A4	12	D	5	down	ff	detached	false
125	A4	7	D	D5	12	D	5	up	mp	detached	true
126	D5	7	D	A4	12	D	5	down	mp	detached	true
127	A4	7	D	D5	12	D	5	up	ff	detached	true
128	D5	7	D	A4	12	D	5	down	ff	detached	true
129	A4	7	D	D5	12	D	5	up	mp	legato	false
130	D5	7	D	A4	12	D	5	down	mp	legato	false
131	A4	7	D	D5	12	D	5	up	ff	legato	false
132	D5	7	D	A4	12	D	5	down	ff	legato	false
133	A4	7	D	D5	12	D	5	up	mp	legato	true
134	D5	7	D	A4	12	D	5	down	mp	legato	true
135	A4	7	D	D5	12	D	5	up	ff	legato	true
136	D5	7	D	A4	12	D	5	down	ff	legato	true
137	A4	7	D	D5	12	D	5	up	mp	glissando	false
138	D5	7	D	A4	12	D	5	down	mp	glissando	false
139	A4	7	D	D5	12	D	5	up	ff	glissando	false
140	D5	7	D	A4	12	D	5	down	ff	glissando	false
141	A4	7	D	D5	12	D	5	up	mp	glissando	true
142	D5	7	D	A4	12	D	5	down	mp	glissando	true
143	A4	7	D	D5	12	D	5	up	ff	glissando	true
144	D5	7	D	A4	12	D	5	down	ff	glissando	true
145	E5	7	A	A5	12	A	5	up	mp	detached	false
146	A5	7	A	E5	12	A	5	down	mp	detached	false
147	E5	7	A	A5	12	A	5	up	ff	detached	false
148	A5	7	A	E5	12	A	5	down	ff	detached	false
149	E5	7	A	A5	12	A	5	up	mp	detached	true
150	A5	7	A	E5	12	A	5	down	mp	detached	true
151	E5	7	A	A5	12	A	5	up	ff	detached	true
152	A5	7	A	E5	12	A	5	down	ff	detached	true
153	E5	7	A	A5	12	A	5	up	mp	legato	false
154	A5	7	A	E5	12	A	5	down	mp	legato	false
155	E5	7	A	A5	12	A	5	up	ff	legato	false
156	A5	7	A	E5	12	A	5	down	ff	legato	false
157	E5	7	A	A5	12	A	5	up	mp	legato	true
158	A5	7	A	E5	12	A	5	down	mp	legato	true
159	E5	7	A	A5	12	A	5	up	ff	legato	true
160	A5	7	A	E5	12	A	5	down	ff	legato	true
161	E5	7	A	A5	12	A	5	up	mp	glissando	false
162	A5	7	A	E5	12	A	5	down	mp	glissando	false
163	E5	7	A	A5	12	A	5	up	ff	glissando	false
164	A5	7	A	E5	12	A	5	down	ff	glissando	false
165	E5	7	A	A5	12	A	5	up	mp	glissando	true
166	A5	7	A	E5	12	A	5	down	mp	glissando	true
167	E5	7	A	A5	12	A	5	up	ff	glissando	true

APPENDIX II. TWO-NOTE LIST

168	A5	7	A	E5	12	A	5	down	ff	glissando	true
169	B5	7	E	E6	12	E	5	up	mp	detached	false
170	E6	7	E	B5	12	E	5	down	mp	detached	false
171	B5	7	E	E6	12	E	5	up	ff	detached	false
172	E6	7	E	B5	12	E	5	down	ff	detached	false
173	B5	7	E	E6	12	E	5	up	mp	detached	true
174	E6	7	E	B5	7	E	5	down	mp	detached	true
175	B5	7	E	E6	12	E	5	up	ff	detached	true
176	E6	7	E	B5	12	E	5	down	ff	detached	true
177	B5	7	E	E6	12	E	5	up	mp	legato	false
178	E6	7	E	B5	12	E	5	down	mp	legato	false
179	B5	7	E	E6	12	E	5	up	ff	legato	false
180	E6	7	E	B5	12	E	5	down	ff	legato	false
181	B5	7	E	E6	12	E	5	up	mp	legato	true
182	E6	7	E	B5	12	E	5	down	mp	legato	true
183	B5	7	E	E6	12	E	5	up	ff	legato	true
184	E6	7	E	B5	12	E	5	down	ff	legato	true
185	B5	7	E	E6	12	E	5	up	mp	glissando	false
186	E6	7	E	B5	12	E	5	down	mp	glissando	false
187	B5	7	E	E6	12	E	5	up	ff	glissando	false
188	E6	7	E	B5	12	E	5	down	ff	glissando	false
189	B5	7	E	E6	12	E	5	up	mp	glissando	true
190	E6	7	E	B5	12	E	5	down	mp	glissando	true
191	B5	7	E	E6	12	E	5	up	ff	glissando	true
192	E6	7	E	B5	12	E	5	down	ff	glissando	true
193	D4	7	G	D4	7	G	0	down	mp	detached	false
194	D4	7	G	D4	7	G	0	down	ff	detached	false
195	D4	7	G	D4	7	G	0	down	mp	detached	true
196	D4	7	G	D4	7	G	0	down	ff	detached	true
197	D4	7	G	D4	7	G	0	down	mp	legato	false
198	D4	7	G	D4	7	G	0	down	ff	legato	false
199	A4	7	D	A4	7	D	0	down	mp	detached	false
200	A4	7	D	A4	7	D	0	down	ff	detached	false
201	A4	7	D	A4	7	D	0	down	mp	detached	true
202	A4	7	D	A4	7	D	0	down	ff	detached	true
203	A4	7	D	A4	7	D	0	down	mp	legato	false
204	A4	7	D	A4	7	D	0	down	ff	legato	false
205	E5	7	A	E5	7	A	0	down	mp	detached	false
206	E5	7	A	E5	7	A	0	down	ff	detached	false
207	E5	7	A	E5	7	A	0	down	mp	detached	true
208	E5	7	A	E5	7	A	0	down	ff	detached	true
209	E5	7	A	E5	7	A	0	down	mp	legato	false
210	E5	7	A	E5	7	A	0	down	ff	legato	false
211	B5	7	E	B5	7	E	0	down	mp	detached	false
212	B5	7	E	B5	7	E	0	down	ff	detached	false
213	B5	7	E	B5	7	E	0	down	mp	detached	true

APPENDIX II. TWO-NOTE LIST

214	B5	7	E	B5	7	E	0	down	ff	detached	true
215	B5	7	E	B5	7	E	0	down	mp	legato	false
216	B5	7	E	B5	7	E	0	down	ff	legato	false
217	D4	7	G	G3	0	G	7	down	mp	detached	false
218	G3	7	G	D4	0	G	7	up	mp	detached	false
219	D4	7	G	G3	0	G	7	down	ff	detached	false
220	G3	7	G	D4	0	G	7	up	ff	detached	false
221	D4	7	G	G3	0	G	7	down	mp	legato	false
222	G3	7	G	D4	0	G	7	up	mp	legato	false
223	D4	7	G	G3	0	G	7	down	ff	legato	false
224	G3	7	G	D4	0	G	7	up	ff	legato	false
225	D4	7	G	G3	0	G	7	down	mp	legato	true
226	G3	7	G	D4	0	G	7	up	mp	legato	true
227	D4	7	G	G3	0	G	7	down	ff	legato	true
228	G3	7	G	D4	0	G	7	up	ff	legato	true
229	D4	7	G	G3	0	G	7	down	mp	glissando	false
230	G3	7	G	D4	0	G	7	up	mp	glissando	false
231	D4	7	G	G3	0	G	7	down	ff	glissando	false
232	G3	7	G	D4	0	G	7	up	ff	glissando	false
233	D4	7	G	G3	0	G	7	down	mp	glissando	true
234	G3	7	G	D4	0	G	7	up	mp	glissando	true
235	D4	7	G	G3	0	G	7	down	ff	glissando	true
236	G3	7	G	D4	0	G	7	up	ff	glissando	true
237	A4	7	D	D4	0	D	7	down	mp	detached	false
238	D4	7	D	A4	0	D	7	up	mp	detached	false
239	A4	7	D	D4	0	D	7	down	ff	detached	false
240	D4	7	D	A4	0	D	7	up	ff	detached	false
241	A4	7	D	D4	0	D	7	down	mp	legato	false
242	D4	7	D	A4	0	D	7	up	mp	legato	false
243	A4	7	D	D4	0	D	7	down	ff	legato	false
244	D4	7	D	A4	0	D	7	up	ff	legato	false
245	A4	7	D	D4	0	D	7	down	mp	legato	true
246	D4	7	D	A4	0	D	7	up	mp	legato	true
247	A4	7	D	D4	0	D	7	down	ff	legato	true
248	D4	7	D	A4	0	D	7	up	ff	legato	true
249	A4	7	D	D4	0	D	7	down	mp	glissando	false
250	D4	7	D	A4	0	D	7	up	mp	glissando	false
251	A4	7	D	D4	0	D	7	down	ff	glissando	false
252	D4	7	D	A4	0	D	7	up	ff	glissando	false
253	A4	7	D	D4	0	D	7	down	mp	glissando	true
254	D4	7	D	A4	0	D	7	up	mp	glissando	true
255	A4	7	D	D4	0	D	7	down	ff	glissando	true
256	D4	7	D	A4	0	D	7	up	ff	glissando	true
257	E5	7	A	A4	0	A	7	down	mp	detached	false
258	A4	7	A	E5	0	A	7	up	mp	detached	false
259	E5	7	A	A4	0	A	7	down	ff	detached	false

APPENDIX II. TWO-NOTE LIST

260	A4	7	A	E5	0	A	7	up	ff	detached	false
261	E5	7	A	A4	0	A	7	down	mp	legato	false
262	A4	7	A	E5	0	A	7	up	mp	legato	false
263	E5	7	A	A4	0	A	7	down	ff	legato	false
264	A4	7	A	E5	0	A	7	up	ff	legato	false
265	E5	7	A	A4	0	A	7	down	mp	legato	true
266	A4	7	A	E5	0	A	7	up	mp	legato	true
267	E5	7	A	A4	0	A	7	down	ff	legato	true
268	A4	7	A	E5	0	A	7	up	ff	legato	true
269	E5	7	A	A4	0	A	7	down	mp	glissando	false
270	A4	7	A	E5	0	A	7	up	mp	glissando	false
271	E5	7	A	A4	0	A	7	down	ff	glissando	false
272	A4	7	A	E5	0	A	7	up	ff	glissando	false
273	E5	7	A	A4	0	A	7	down	mp	glissando	true
274	A4	7	A	E5	0	A	7	up	mp	glissando	true
275	E5	7	A	A4	0	A	7	down	ff	glissando	true
276	A4	7	A	E5	0	A	7	up	ff	glissando	true
277	B5	7	E	E5	0	E	7	down	mp	detached	false
278	E5	7	E	B5	0	E	7	up	mp	detached	false
279	B5	7	E	E5	0	E	7	down	ff	detached	false
280	E5	7	E	B5	0	E	7	up	ff	detached	false
281	B5	7	E	E5	0	E	7	down	mp	legato	false
282	E5	7	E	B5	0	E	7	up	mp	legato	false
283	B5	7	E	E5	0	E	7	down	ff	legato	false
284	E5	7	E	B5	0	E	7	up	ff	legato	false
285	B5	7	E	E5	0	E	7	down	mp	legato	true
286	E5	7	E	B5	0	E	7	up	mp	legato	true
287	B5	7	E	E5	0	E	7	down	ff	legato	true
288	E5	7	E	B5	0	E	7	up	ff	legato	true
289	B5	7	E	E5	0	E	7	down	mp	glissando	false
290	E5	7	E	B5	0	E	7	up	mp	glissando	false
291	B5	7	E	E5	0	E	7	down	ff	glissando	false
292	E5	7	E	B5	0	E	7	up	ff	glissando	false
293	B5	7	E	E5	0	E	7	down	mp	glissando	true
294	E5	7	E	B5	0	E	7	up	mp	glissando	true
295	B5	7	E	E5	0	E	7	down	ff	glissando	true
296	E5	7	E	B5	0	E	7	up	ff	glissando	true
297	D4	7	G	A4	7	D	7	up	mp	detached	false
298	A4	7	G	D4	7	D	7	down	mp	detached	false
299	D4	7	G	A4	7	D	7	up	ff	detached	false
300	A4	7	G	D4	7	D	7	down	ff	detached	false
301	D4	7	G	A4	7	D	7	up	mp	detached	true
302	A4	7	G	D4	7	D	7	down	mp	detached	true
303	D4	7	G	A4	7	D	7	up	ff	detached	true
304	A4	7	G	D4	7	D	7	down	ff	detached	true
305	D4	7	G	A4	7	D	7	up	mp	legato	false

APPENDIX II. TWO-NOTE LIST

306	A4	7	G	D4	7	D	7	down	mp	legato	false
307	D4	7	G	A4	7	D	7	up	ff	legato	false
308	A4	7	G	D4	7	D	7	down	ff	legato	false
309	D4	7	G	A4	7	D	7	up	mp	legato	true
310	A4	7	G	D4	7	D	7	down	mp	legato	true
311	D4	7	G	A4	7	D	7	up	ff	legato	true
312	A4	7	G	D4	7	D	7	down	ff	legato	true
313	A4	7	D	E5	7	A	7	up	mp	detached	false
314	E5	7	D	A4	7	A	7	down	mp	detached	false
315	A4	7	D	E5	7	A	7	up	ff	detached	false
316	E5	7	D	A4	7	A	7	down	ff	detached	false
317	A4	7	D	E5	7	A	7	up	mp	detached	true
318	E5	7	D	A4	7	A	7	down	mp	detached	true
319	A4	7	D	E5	7	A	7	up	ff	detached	true
320	E5	7	D	A4	7	A	7	down	ff	detached	true
321	A4	7	D	E5	7	A	7	up	mp	legato	false
322	E5	7	D	A4	7	A	7	down	mp	legato	false
323	A4	7	D	E5	7	A	7	up	ff	legato	false
324	E5	7	D	A4	7	A	7	down	ff	legato	false
325	A4	7	D	E5	7	A	7	up	mp	legato	true
326	E5	7	D	A4	7	A	7	down	mp	legato	true
327	A4	7	D	E5	7	A	7	up	ff	legato	true
328	E5	7	D	A4	7	A	7	down	ff	legato	true
329	E5	7	A	B5	7	E	7	up	mp	detached	false
330	B5	7	A	E5	7	E	7	down	mp	detached	false
331	E5	7	A	B5	7	E	7	up	ff	detached	false
332	B5	7	A	E5	7	E	7	down	ff	detached	false
333	E5	7	A	B5	7	E	7	up	mp	detached	true
334	B5	7	A	E5	7	E	7	down	mp	detached	true
335	E5	7	A	B5	7	E	7	up	ff	detached	true
336	B5	7	A	E5	7	E	7	down	ff	detached	true
337	E5	7	A	B5	7	E	7	up	mp	legato	false
338	B5	7	A	E5	7	E	7	down	mp	legato	false
339	E5	7	A	B5	7	E	7	up	ff	legato	false
340	B5	7	A	E5	7	E	7	down	ff	legato	false
341	E5	7	A	B5	7	E	7	up	mp	legato	true
342	B5	7	A	E5	7	E	7	down	mp	legato	true
343	E5	7	A	B5	7	E	7	up	ff	legato	true
344	B5	7	A	E5	7	E	7	down	ff	legato	true

Configuration File

```
# GL000 Settings
#
# This file is passed to the GL000_synth for startup:
#
# - comment or uncomment lines to change
#   the general synth configuration
# - set sepcific values to change behaviour
#
# Henrik von Coler
# 2019-11-21

#####
# Paths and config files
#####

# The standard assignment:
# assignment_file: "sample-assignment_STANDARD.txt"

# A minimized version for debugging:
assignment_file: "sample-assignment_SMALL.txt"

bark_file: "bark-bank_48000.yml"

#####
# Basic synth settings.
#####

# delta T used for stepping though the sinusoidal data
sms_rate: 0.01

# maximum number of simultaneously running voices
num_voices: 4

# the maximum number of partials in the data set:
num_partials: 80

# detrmine which data sets to read
synth_mode: "dual"
#synth_mode: "deterministic"
#synth_mode: "stochastic"

# ATTACK and RELEASE modes
```

```

#attack_mode: "original"
attack_mode: "linear"
release_mode: "fixed"
# release_mode: "statistic"

#sine_mode: "lookup"
sine_mode: "cmath"

#####
# OSC settings
#####

osc_port_in: 5100          # from Bong
osc_port_out: 1234        # to SSR
receive_osc: 1
receive_midi: 1

#####
# DEBUG outupts and alike
#####

OUTPUTS:
    # set '1' to print all amplitudes to txt-files
    partial_trajectories: 0

    # chose path to create txt files
    partial_output_path: "../debug_outputs/"

    # display incoming performance controls in terminal
    control_input: 0

    # display information regarding the stochastic synthesis
    # on the terminal
    statistical_data: 0

```