Eden: Visualization Techniques and Sound Synthesis based on Oscilloscope Art

Mathias Lehnfeld, MSc

MASTER THESIS

Sound Design

supervised by
Univ.Prof. Mag. Ph.D. Marko Ciciliani
University of Music and Performing Arts Graz

head of degree program
Prof. Dr. Josef Gründler
FH Joanneum, University of Applied Sciences

June 2017

Declaration

Ich erkläre ehrenwörtlich, dass ich die vorliegende Masterarbeit selbstständig angefertigt und die mit ihr verbundenen Tätigkeiten selbst erbracht habe. Ich erkläre weiters, dass ich keine anderen als die angegebenen Hilfsmittel benutzt habe. Alle aus gedruckten, ungedruckten oder dem Internet im Wortlaut oder im wesentlichen Inhalt übernommenen Formulierungen und Konzepte sind gemäß den Regeln für gutes wissenschaftliches Arbeiten zitiert und durch Fußnoten bzw. durch andere genaue Quellenangaben gekennzeichnet.

Die vorliegende Originalarbeit ist in dieser Form zur Erreichung eines akademischen Grades noch keiner anderen Hochschule vorgelegt worden. Diese Arbeit wurde in gedruckter und elektronischer Form abgegeben. Ich bestätige, dass der Inhalt der digitalen Version vollständig mit dem der gedruckten Version übereinstimmt. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben kann.

Graz, am 19. Juni 2017

Mathias Lehnfeld, MSc

Contents

Acknowledgements

First and foremost, my gratitude belongs to Jethro Cooke who not only is the author of the amazing audiovisual performance that inspired this project. He also eagerly explained the internal mechanisms, answered all my questions and allowed me to reuse his techniques.

Secondly, I want to thank my thesis supervisor Marko Ciciliani. His answers and feedback were always as helpful as they were constructive and motivating. He agreed to guide my work although I was not personally in Graz and despite his busy schedule put up with my fickle time management. Furthermore, his seminars were some of the most instructive and revealing courses I have had the chance to attend.

Also, I am very grateful to Josef Gründler. Besides always being available for my technical and organizational questions, he encouraged everyone to follow their interests throughout the program. I very much appreciated the comfortable, personal and inspiring atmosphere at the university.

My colleagues also deserve my gratitude. Their competent and collegial personalities have made the last two years a very enjoyable time. It is impossible to overrate how much they have taught me. My special thanks go to Benni and Seidl for patiently answering 101 questions per day, being most easy-going and fun flatmates and for their input to this project.

Finally, I want to express my thanks to Simon for helping me create and produce the video.

For Jakob, my nephew and godson.

Abstract

Analog oscilloscopes in X-Y mode visualizing stereo audio signals have previously been used for artistic purposes. Digital implementations enable additional layers of creative exploration such as the introduction of colors or layering the output of multiple oscilloscopes. Thereby, the connection of image and sound can be extended and the audiovisual experience potentially enhanced. In this context, the thesis documents the prototypical development of a sound synthesis and visualization technique and a performance which incorporates several possible means of using it for artistic expression. In addition, the technical foundations are established and related topics such as psychology of perception and other projects that inspired Eden are discussed.

Kurzfassung

Analoge Oszilloskope im XY-Modus fanden in Verbindung mit Audiosignalen bereits künstlerischen Einsatz. Digitale Umsetzungen ermöglichen weitere Gestaltungsebenen wie die Verwendung von Farbe oder eine Überlagerung der Ausgabe mehrerer Oszilloskope, welche die Verbindung von Bild und Ton erweitern und so das audiovisuelle Erlebnis bereichern können. In diesem Kontext begleitet die vorliegende Arbeit die exemplarische Umsetzung einer Klangsynthese- und Visualierungtechnik, sowie einer Performance, welche einige mögliche Methoden zum künstlerischen Einsatz kombiniert. Damit verbundene wahrnehmungspsychologische Grundlagen werden ebenso diskutiert wie technische Hintergründe und weitere Projekte, welche die vorliegende Arbeit inspiriert haben.

Chapter 1

Introduction

Digital audiovisual art has a long history. During the last decades, developments in computer processing power, graphics hardware and the evolution of software has opened up a new world of possibilities for the creation of digital audiovisual art. Audio reactive visuals are based on audio input. A vast amount of properties can be extracted from an audio signal and mapped to visual parameters in real-time.

Contributors to the computer art subculture *demoscene* focus on showing off technical, artistic and musical skills. The majority of works that use audio streams directly to generate visuals, like oscilloscope art, offer audiovisual impressions that may be aesthetically pleasing and through congruencies of certain properties of image and sounds they usually create integrated experiences that are more than the perceptions on two independent levels.

Inspired by a performance of *Asterism* by Jethro Cooke, this thesis will establish the theoretical background and follow the development and exemplary implementation of an audio visualization technique in which the application of an audio signal processor yields a visual output on a digital XY-oscilloscope. In addition to results that can be achieved on analog oscilloscopes, a number of extensions exploit the advantages of digital technology to enable new layers of connections between image and sound such as mapping spectral information to the color dimension or overlaying the outputs of more than one oscilloscope simultaneously. Artifacts caused by limited bit depth and sample rate in the digital domain may be considered a disadvantage over analog systems. From an artistic point of view they can be utilized to intentionally create certain visual effects. Finally, a composition designed to demonstrate a number of ways in which the technology can be used artistically investigates the possibilities and limits of the visualization mechanism in an attempt to generate a coherent combination of image and sound.

1.1 Motivation

The techniques used in the production of *Asterism* are related to oscilloscope art. In the course of an email conversation, Cooke explained the process in detail to me. The mechanism which I would like to call *Wave Terrain Distortion* is an audio signal processor with one input channel and one output channel. The visualization then interprets input samples as a stream of x-coordinates and the output samples as the according y-coordinates which results in lines and shapes on a two dimensional space. This is exactly what an oscilloscope in XY-mode is designed to accomplish, albeit for a functional purpose rather than an artistic one. In the context of a semester project I explored oscilloscope art, notably by Jerobeam Fenderson, where sounds are designed specifically for the purpose of creating certain images on an oscilloscope. In some cases, maintaining auditory aesthetics is a priority as well. The revision of Fenderson's productions sparked an excitement for the subject matter in me. In this thesis I expand on his and Cooke's works and establish a personal artistic angle and approach to an existing technique for audiovisual performance.

1.2 Objective

My vision for the outcome of this project is a combination of a visualization algorithm with ways for audio synthesis that creates simple and obvious connections between auditory and visual experiences and, as a consequence, new, multimodal qualities. The final prototype shall translate sounds into images in a nuanced and versatile way and thereby make it possible to create numerous performances with different kinds of audio input signals so that the resulting visuals and movements show distinct and heterogenous patterns. This means that the technique should maintain a certain degree of congruence. At the same time audiovisual relationships should not be redundant or predictable but rather express surprising elements that are too complex to anticipate. It should also be feasible to maintain a certain degree of aesthetic quality in both, audio and visual outcomes. I hope that this project inspires experiments with various synthetic and physical sounds and parameters and serves as a source of ideas for extensions and modifications, thereby giving way to other audiovisual performances.

1.3 Document Structure

The next chapter provides an introduction to audiovisual perception and tuning systems based on just intonation, both to the extent relevant for this project. In chapter 3 we will get a brief overview on the history of music visualization and oscilloscope art. Chapter 4 deals with the software setup and platform that provided the context in which the prototype was developed. Chapters 5 and 6 then discuss the prototype's technological and artistic details, respectively. Finally, 7 concludes the documentation of the project, emphasizes the central points, critically reflects the outcome and possible extensions and futurework.

Chapter 2

Foundations

2.1 Audiovisual Perception

The processing of sensory information in the human brain is a highly complex and extensively studied area. Multisensory perception investigates how we integrate stimuli from different sensory modes to gain a coherent perception of objects and events in the real world (Recanzone 2009).

There are several timing disparities between sensation and processing of auditory and visual stimuli. Most importantly, sound generally travels about six orders of magnitude slower in air than light does. Thus, acoustic and visual attributes of an event arrive at the sensory organs at different points in time. In contrast, the stimulus transduction and lower level processing in the brain is significantly faster for auditory streams than it is for visual input with primary visual neurons showing higher latencies than primary auditory neurons. This latency difference, however, is not present in multisensory cortical areas which integrate both sensory modes. While it has been shown that there is a tolerance delay for perceiving stimuli from the two modes as occurring simultaneously, experiments have shown that subjects are more likely to perceive auditory and visual stimuli in segregation, when the former are presented before the latter. This result is consistent with simultaneous emission of stimuli in the physical world, where sound cannot possibly arrive at the subject sooner than light (Recanzone 2009, 1 sq.).

Another study concludes that spatial and temporal proximity of visual and auditory stimuli influence each other and thus the integration of the two senses depends on many aspects, even in clinical setups (Recanzone 2009, 2).

The Ventriloquism Effect

The ventriloquism effect has been used to study several aspects of multisensory integration. It describes an auditory illusion where an auditory stimulus is associated with a visual one because the latter dominates the overall perception of an event or object even though it may in fact not be causally related to the auditory stimulus. The markedness of the ventriloquism effect has been shown to depend on three main factors (Recanzone 2009, 3 sqq.):

compellingness means that the movements of the ventriloquist's mouth are minimal and the heard voice is appropriate for the puppet's appearance.

timing is a complex matter in a ventriloquism performance and deals with the quality of the coupling between audible speech and the way the puppet's mouth moves.

spatial proximity between ventriloquist and puppet is crucial.

These considerations are connected in complex ways where the quality of one heavily influences the importance of another. There are hypotheses about cognitive heuristics, i.e. strategies for making sense of and organizing sensations, and how we take into account factors like reliability, acuity and salience to prioritize stimuli and disparities in order to create coherent perceptions (Recanzone 2009, 10).

Our perception of the two modes shows different acuity in different domains. For example, the discrimination of auditory sequence timing is much better than with visual sequences while spatial discrimination is generally superior in the visual system (Recanzone 2009, 10). As a general, certainly scientifically oversimplified, but for our purposes useful conclusion, we can say that when integrating possibly contradictory auditory and visual stimuli, our perception has a tendency to rely on visual stimuli for spatial cues and on the auditory system for temporal information.

2.1.1 Chromesthesia

Synesthesia is a condition where sensations of one sensory system evoke experiences in another one. In the case of chromesthesia, the sensation of sounds causes the preception of color. The actual mappings are highly idiosyncratic, i.e. the characteristics such as timbre, tempo or pitch that result in certain colors and the concrete correlations differ between every individual. Both, synesthetes and nonsynesthetes associate musical attributes with colors. How this occurs is not clear, but the authors of (Palmer, Schloss, Xu, and Prado-León 2013) argue that for nonsynesthetes there is abundant evidence for common tendencies in Western cultures and that associations are mediated by experiences of emotion in relation to music and colors. Their results show a positive correlation between faster music in major keys and light, saturated colors closer to yellow, while slow music in minor keys is statistically related to dark, less saturated colors closer to blue. Timbre and saturation, pitch and size as well as loudness and brightness seem to be linked according to other studies (Caivano 1994; Evans and Treisman 2009; Marks 1987; Parise and Spence 2012), but the results are not representative enough to draw conclusions on reliable, culture invariant principles. Robust positive correlations between pitch and lightness have been shown in several studies in nonsynesthetes as well as synesthetes, i.e. higher pitches tend to be associated with lighter colors and lower pitches with dark colors (Hubbard 1996; Marks 1974, 1987; Ward, Huckstep, and Tsakanikos 2006). A study from 1956 was conducted with synesthetic children and yielded reliable pitch-hue relationships (Simpson, Quinn, and Ausubel 1956). The authors of (Palmer, Schloss, Xu, and Prado-León 2013) attribute these results to the different perceptions of lightness of colors. The following connections were found and we will make use of them in this project (see section 5.2.5): high pitches were associated with yellow and green (light colors), midrange pitches with red and orange (darker) and low pitches with blue and violet (even darker).

2.1.2 Perceived Congruence

Iwamiya distinguishes between different kinds of congruence, which constitute the overall impression of image and sound "belonging together" (Iwamiya 2013).

Formal Congruence This addresses temporal synchronization of visual and auditory elements. Perceived correlation is not only limited to events occurring simultaneously, but can also be experienced on the basis of common patterns. E.g. image and sound could express the same rhythms in the form of certain movements or accents, but there might be a constant temporal offset between them and the correlation might still be perceptible (see Iwamiya 2013, 145 sqq.).

Semantic Congruence The contribution of similarity in affective impressions to the overall perceived congruence is called semantic congruence. Multiple studies have been conducted to investigate the various ways in which image and sound may produce related affective impressions and thereby induce a feeling of congruence between the two media (see Iwamiya 2013, 147 sqq.). With varying degrees of robustness with respect to applicability in everyday life, the correlations listed in table 2.1 were evident. The referred research did not systematically consider psychophysical properties of specific colors, but clearly showed that colors generally intensified affective impressions, regardless of any potential congruence.

It is important to mention that while in practice, real effects of links between auditory and visual properties listed in table 2.1 on the perception of congruence are extremely likely, the simplified experiment setup does not generally allow for reliable conclusions on practical applicability. Nevertheless, considering the results for application is certainly reasonable within the context of this project.

Table 2.1: semantic congruence: evident relationships (Iwamiya 2013, 147 sqq.)

visual	*auditory*
high density, fast	major key, fast tempo
low density, slow	minor key, slow tempo
brighter	major key
darker	minor key
low / high vertical position	low / high pitch
size	loudness
shape	timbre

Changing Patterns Closely related to the previous attributes, some studies (e.g. Arita and Iwamiya 2004; Arita, Su, Kawakami, Ueda, and Iwamiya 2005; Lipscomb and Kim 2004; Maeda, Kanai, and Shimojo 2004; Rusconi, Kwan, Giordano, Umilta, and Butterworth 2006, see Iwamiya 2013, 152 sqq. for summaries) have identified correlations between more dynamic aspects in audiovisual media. For instance, changes in pitch may easily be considered congruent to certain movements in the image. These relationships follow directional patterns and their relevance may be compromised by inverting the correlations, or they may not be significant with arbitrary, synchronized changes. Table 2.2 lists rather reliable connections. It is interesting to realize that many languages, including English, use corresponding terms for “up” and “down” for several domains, most importantly for vertical movement and pitch changes alike. As one would therefore expect in the case of simultaneous movements on both axes in the visual stream, the influence of the vertical direction on the overall perception of congruence is more distinct than the horizontal direction (Kim, Iwamiya, and Kitano 2008).

Table 2.2: changing patterns: robust correlations (Iwamiya 2013, 142 & 152 sqq.)

visual	*auditory*
left to right / upward movement	ascending / high pitch
right to left / downward movement	descending / low pitch
speed	musical tempo
verticality	pitch contour
distance	dynamics

Please note that the studies that yielded these data have not verified, to which degree these conclusions are applicable across different cultures. However, for our artistic purposes we will assume that these observations are likely to enhance the experienced coherence of audio and visuals.

Generally speaking, the presence of one type of congruence improves perception of others, i.e. subjects are more likely to perceive different types of congruence in the presence of other, more salient congruent features (Iwamiya 2013, 161 sq.).

2.1.3 Audiovisual Relationships

On the basis of previous publications, Ciciliani identifies a number of possible ways in which sound and moving images can interact in musical multimedia (Ciciliani 2016). The categories are designed as part of an introspective evaluation of the subjective experience generated by the audiovisual interplay. This is relevant for our purposes because we can build upon the parameters from table 2.3 and consider if, how and to what extent we want to create the according perceptions. Also, it would be interesting to perform this evaluation method on the final outcome. However, this will not be done within the scope of the project.

Table 2.3: analysis parameters established by (Ciciliani 2016)

mapping	*semantics*	*atmosphere*
synchrony	congruence of content	kinetic congruence
spatial congruence	coherence of idiom	balance of salience
congruence of mass		tinting correspondence

The eight factors listened in the table are ones of activity and are assessed on a scale between divergent and congruent, while the three categories *mapping*, *semantics* and *atmosphere* relate to potency, i.e. they are judged in terms of aesthetic relevance. Ciciliani suggests determining the activity factors first and then proceeding to the potency categories. He argues that there is no immediate correlation between the potency of a group of features and its individual parameters because they may not necessarily be what characterizes the aesthetic significance of the category. Finally, the valence addresses the overall degree of coherence which is not supposed to be a judgment of the work at hand, but rather a subjective, non-judgmental evaluation of how image and sound interact. It is important to emphasize that despite evidence from a music psychology perspective there is no reason to generally strive for congruence over divergence. Artists may explicitly target the perception of divergence in order to challenge the audience or avoid loss of attention due to persistent congruence. Let us now take a closer look at the parameters.

Mapping

The lower level mappings are somewhat based on many of the observations we discussed in the previous section.

synchrony is concerned with temporal congruence and generally used to create a sense of cohesion between sound and image.

spatial congruence refers to the location, movement and expansion of visual elements and their relation to sounds, their panning behavior and pitch related characteristics.

congruence in terms of mass describes the extent to which the perceived weight or size of auditory phenomena relates to the visual domain. Note that some ways in which properties may contribute to the perception of congruence of this kind were discussed in the previous section.

Semantics

This category takes the evaluation to a higher cognitive level. It is concerned with the perception of meaning.

congruence of content can arise from the matching of narrative context, content-related and symbolic references or whenever the combination of image and sound may create a new meaning present in neither stream independently.

coherence of idiom is present when idiomatic or stylistic approaches in sonic and visual domains match, as opposed to e.g. 8-bit music alongside scenes of an 18th century opera.

Atmosphere

The following three parameters are concerned with the overall atmosphere created by image and sound. This goes beyond relationships between single events or elements on either domain, towards the higher-level character of single scenes and whole pieces of work.

kinetic congruence is somewhat similar to the synchrony parameter but considers the temporal relationships of the different media on a bigger scale, not on the level of synchronicity between single events but rather in terms of a general sense of speed or pace.

balance of salience indicates whether the two media dominate each other. Another aspect of this parameter regards the extent to which the levels of detail expression are coherent.

correspondence of tinting addresses the coherence of moods generated by image and sound.

2.2 Tunings Based on Just Intonation

12-tone equal temperament has been the most common tuning system in Western music since the nineteenth century. It splits every octave into twelve semitones with a constant frequency ratio of $\sqrt[12]{2}$ (Hinrichsen 2016). Consequently, it is independent from musical keys. This, however, is a compromise since intervals in just intonation can only be approximated when a tuning system is based on constant intervals. Instead, they are based on the harmonic series, which is the sequence of frequencies that are integer multiples of any arbitrary reference frequency f_0. The vibrations occurring in musical instruments express different frequencies of the harmonic series at characteristic amplitudes, leading to the perception of a certain timbre. The series $(2^1\,f_0, 2^2\,f_0, ...)$ contains all octaves higher than the base note. The just frequency ratios for other musical intervals result from other multiples. E.g. the third harmonic $3\,f_0$ is one octave plus a perfect fifth higher than the fundamental note (Honingh 2006, 7-13). From this follows the ratio $\frac{3}{2} = 1.5$ for a perfect fifth. In equal temperament tuning, the ratio for a fifth is $2^{7/12} \approx 1.4983$. Similarly, a just major third equals $\frac{5}{4} = 1.25$ as opposed to $2^{4/12} \approx 1.2599$ in 12-tone equal temperament.

For our visualization, the exact intervals occurring in the audio input signal will be relevant. It is essential that the frequency ratios are rational numbers with small coefficients because their least common multiple and the base frequency determine the length of the periodic path on the terrain and thereby whether it can create the perception of a still figure or rather chaotic, ever-changing shapes.

We have seen that a just tuning system is defined on the basis of a reference frequency. Once the exact ratios $(r_0, r_1, ..., r_{11})$ for all intervals in an octave are determined, any note is defined by $2^p\,r_n$ where p is the positive or negative number of octaves from the reference note.

5-limit tuning refers to a tuning system where the interval ratios are made up from prime factors no higher than five (Erlich 1998). While less common than 5-limit tuning, 7-limit tuning allows fractions with even smaller coefficients by also permitting 7 as a prime factor. Table 2.4 lists the exact intervals used in the implementation (see fig. A.3 in A).

Table 2.4: 7-limit tuning ratios. Source: https://en.wikipedia.org/wiki/Five-limit_tuning#The_justest_ratios

interval name	*semitones*	*ratio*
perfect unison	0	1:1
minor second	1	15:14
major second	2	9:8
minor third	3	6:5
major third	4	5:4
perfect fourth	5	4:3
augmented fourth	6	7:5
diminished fifth	6	10:7 (omitted)
perfect fifth	7	3:2
minor sixth	8	8:5
major sixth	9	5:3
minor seventh	10	7:4
major seventh	11	15:8

Chapter 3

Audio Visualization

This chapter will take a look at historical approaches towards music visualization and previous projects in the context of oscilloscope art. For our survey of the history of music visualization we will focus on those examples that are based on analog or digital signal processing. These will be more tangible for the reflections on this project than other notable ideas and experiments, such as the *Clavecin Oculare*, a light organ invented by Louis-Bertrand Castel during the 18th century (e.g. Peacock 1988).

3.1 History of Music Visualization

The first music visualizer was an exclusively analog console named *Video Music* which the company *Atari* released in 1976 (Fourney and Fels 2009). Designed by Robert Brown, who also invented the classic *Pong* game, it was intended to be used with a record player and a TV. The resulting visuals showed combinations of diamonds in various colors. This was the precursor to visualizers later included with software audio players.

Already in 1979, Mitroo et al. suggested and implemented a system for software based real time visualizations based on the assumption that harmonic relationships could be expressed by using colors similarly to musical notes (Fourney and Fels 2009).

Nowadays, many software media players ship with visualizer plugins. They make use of frequency and beat information extracted from audio streams. For example, *Advanced Visualization Studio* is a plugin for Winamp which enables users to develop their own visualizations with an integrated scripting language (Stine).

Strictly speaking, audio data visualizations intended for analysis purposes may also be considered audio reactive visuals. While they shall not be the focus of our investigation, we may still be able to gain value from them because not only do they provide elaborate means for feature extraction, but they also give us sensible, proven mappings from auditory features to visual properties.

Music education is another area where visual feedback has been employed. Analysis systems have been developed in support of musicians working on their sound (e.g. Ferguson, Moere, and Cabrera 2005).

More recently, we have seen a vast number of new software projects that strive to make the creation of individual real time music visualizations as easy as possible. Examples include desktop software such as *Synesthesia*[1], *Magic Music Visuals*[2], *VSXu*[3], *Plane9*[4], and also thanks to recent advances in Web Audio and WebGL technologies, web based solutions like *Butter Churn*[5] and *APEXvj*[6]. In addition, there is an abundance of systems for audiovisual programming on the desktop (e.g. *Pure Data* and *Gem* (see section 4.2), *Max/MSP* and *Jitter*, *Processing*, *vvvv*[7]) and JavaScript libraries (e.g. *three.js*[8], *p5.js*[9]) and even browser based development environments (e.g. *Gibber*[10]) that make the generation of real time 2d and 3d graphics very simple and thus accessible for artists who do not necessarily have extensive experiences in computer programming.

3.2 Oscilloscope Art

The ability to use oscilloscopes in XY-mode has inspired a number of projects. Generally, oscilloscopes display changes of electrical signals over time. In the context of audio signals this means that they usually display wave shape data. In XY-mode, two different signals determine the respective coordinates on the two axes, respectively, at any given point in time. This does not only allow for interesting figures resulting from simple, periodic waves (see section 3.2.1). Because any position in the two-dimensional coordinate system can be targeted by the input, anything can be drawn. The only limitation with analog oscilloscopes is that the input voltages must be continuous and thus, jumps on either axis are not possible. This is especially a problem when we generate a signal digitally at a low sample rate. Since the sample rate establishes an upper bound on the maximum frequency that can be processed

[1]"Synesthesia - Live Music Visualizer - VJ Software" 2017.

[2]"Music Visualizer, VJ Software & Beyond: Magic Music Visuals" 2017.

[3]"VSXu - audio visualizer, music visualizer, visual programming language (VPL), realtime graphics design platform" 2017.

[4]"Plane9 a scene based music visualizer and sound responsive screensaver" 2017.

[5]"Butterchurn visualizer" 2017.

[6]"APEXvj = Good vibes" 2017.

[7]"vvvv - a multipurpose toolkit | vvvv" 2017.

[8]"three.js - Javascript 3D library" 2017.

[9]"p5.js | home" 2017.

[10]"Gibber" 2017.

according to the Nyquist-Shannon sampling theorem (Shannon 1949), quick changes on one axis will be smoothed out by the digital to analog converter, thereby reducing the maximum speed at which the voltage on the input to the oscillator can possibly change. In practice, the limitation of continuous signals on an analog level is not an issue if independent visual structures need to be displayed because the achievable speeds are high enough for the human eye not to be able to perceive the connections between them.

The first analog video game was invented in 1958 and it was played on an oscilloscope (Higinbotham 2017). Even the computer game *Quake* was ported to an oscilloscope in late 2014 (Väänänen 2014).

3.2.1 Lissajous Figures

Lissajous figures were first investigated in 1815 by Nathaniel Bowditch, an American mathematician, and are therefore also called *Bowditch curves*. Their more common name comes from Jules-Antoine Lissajous, a French mathematician, who studied them more than forty years later with the help of sand and a compound pendulum (Encyclopædia Britannica 2017). The figures occur, when two harmonic waves are combined. Fig. 3.1 shows a few examples of figures generated by sine waves.

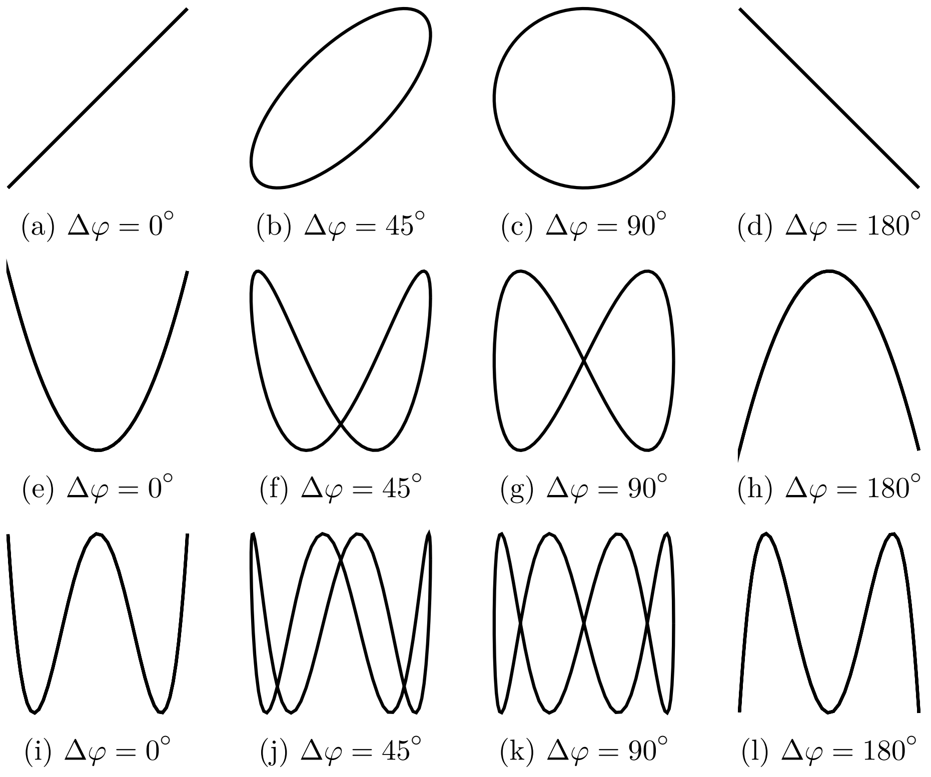
(a) $\Delta\varphi = 0°$
(b) $\Delta\varphi = 45°$
(c) $\Delta\varphi = 90°$
(d) $\Delta\varphi = 180°$
(e) $\Delta\varphi = 0°$
(f) $\Delta\varphi = 45°$
(g) $\Delta\varphi = 90°$
(h) $\Delta\varphi = 180°$
(i) $\Delta\varphi = 0°$
(j) $\Delta\varphi = 45°$
(k) $\Delta\varphi = 90°$
(l) $\Delta\varphi = 180°$

Figure 3.1: Examples of Lissajous figures. The ratio of the two oscillator's frequencies $\frac{f_y}{f_x}$ is 1 for (a)-(d), 2 for (e)-(h) and 4 for (i)-(l).

3.2.2 Oscilloscope Music

Christian Ludwig is an electronic music artist from Austria who performs under the synonym *Jerobeam Fenderson* (Patel 2015). In June 2015, Fenderson launched a Kickstarter campaign to fund the production of a music album titled *Oscilloscope Music*. The campaign was successful and resulted in a number of audio tracks that produce stunning visuals on an oscilloscope. Fenderson's ambition is to create tracks that do not only look impressive on an oscilloscope in XY-mode, but that they are also pleasing to listen to (Fenderson 2017a). See fig. 3.2 for an example.

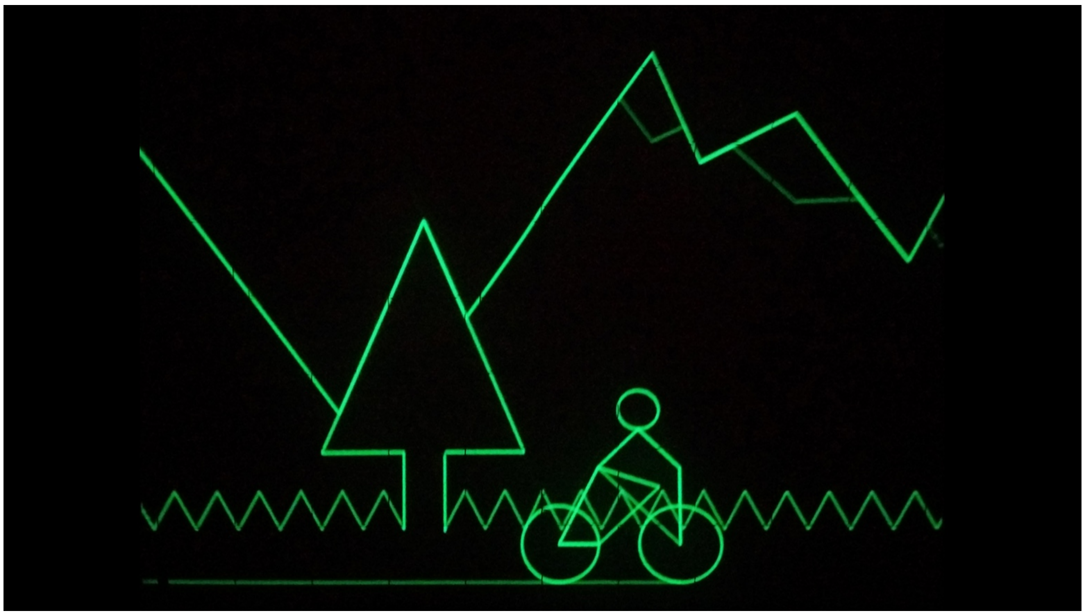

Figure 3.2: Screenshot from Jerobeam Fenderson's *Oscilloscope Music* Kickstarter video. Source: (Fenderson 2017a)

On his blog (Fenderson 2015), Fenderson has published a few Pure Data patches that pack certain functionalities he incorporates into his videos in which he captures the output of his analog Tektronix D11 5103N oscilloscope. His most advertised visualizations include mushrooms and butterflies. Most of the time he works in Max and Pure Data. Some of his more complex performances use a piece of software by Hansi Raber called *OsciStudio*. It comes with a plug-in for the 3D modeling software *Blender* and generates the sounds which in turn produce the 3d data that were previously designed in Blender, when used as input to an oscilloscope. The graphic possibilities this technology allows for seem unlimited. E.g. because the Kickstarter campaign reached more than the initial funding goal, Fenderson published a bonus track in stereoscopic 3D for anaglyph 3D glasses. This is accomplished by overlaying the oscilloscope outputs of two stereo signals, one in cyan and another one in magenta, corresponding to two different camera positions which account for the viewing angle offset of the human eyes (Fenderson 2017b).

3.2.3 Asterism

Asterism is a piece developed and performed by Jethro Cooke. The second version is available on Vimeo (Cooke 2016), where it was uploaded in early 2016. While there is no information on the involved mechanisms available on the web, Cooke kindly explained the process to the author in great detail. *Asterism* was the main source of inspiration for the project documented in this thesis. This section will only give a rough overview of how it was accomplished. The tools he used were different, but for the project's purposes I implemented the same mechanisms and expanded on them. The relevant details can be found in chapter 5.

Cooke works with *Ableton Live* to synthesize the input sounds. What follows is an audio signal processor which is based on concepts from wave terrain synthesis (see section 5.2.1). The audio input provides the x-axis data for an oscilloscope, the processed and delayed audio output the according y-component. The implementation was done with *Max/MSP* and *Jitter* (Cooke, pers. comm.). Section 5.2.2 explains the details of this process. Despite its simplicity and elegance, it yields most impressive visuals (fig. 3.3).

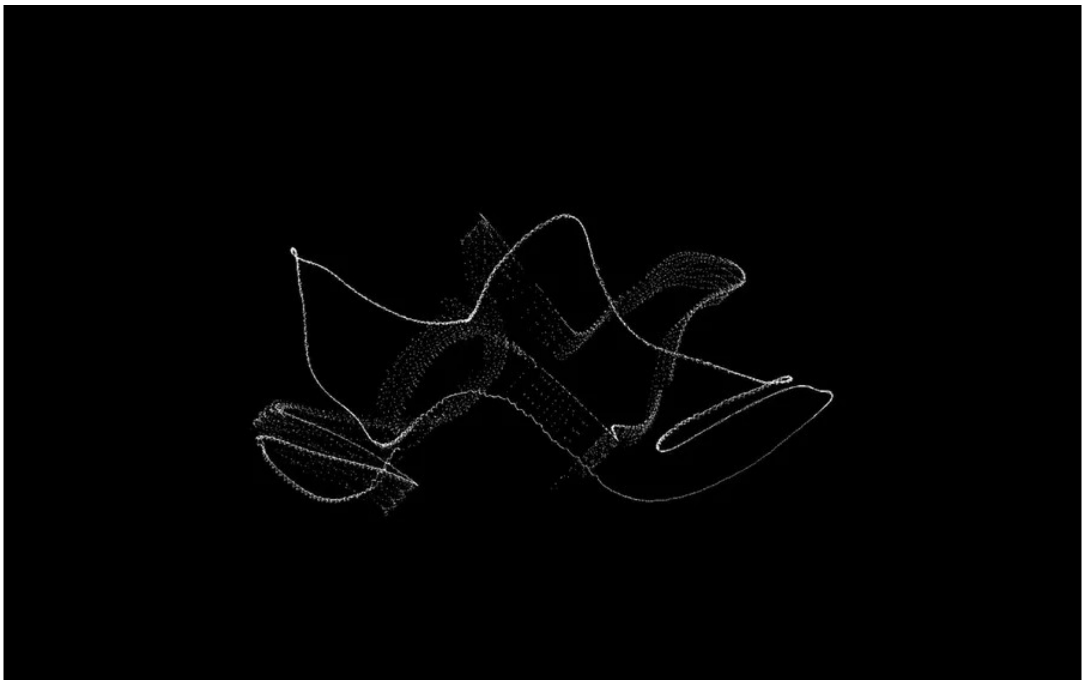

Figure 3.3: Screenshot from Jethro Cooke's *Asterism 2.0* video. Source: (Cooke 2016)

Chapter 4

Software Environment

Having used GNU/Linux exclusively for several years and in addition being an open source advocate, some software options were out of the question for me, most notably Ableton Live and Max/MSP/Jitter which is the setup Jethro Cooke used for *Asterism*. As a software developer, my first choice were textual programming languages. However, Pure Data turned out to be the best tool for my purposes for reasons we will discuss in the following section.

The GNU/Linux distribution platform was an Arch Linux installation with a 4.11 preemptive kernel. The audio server was JACK version 1.9.10 which allowed for audio and MIDI routing between devices and applications. For the production of the final video, I used the digital audio workstation Ardour version 5.9.

4.1 Previous Prototypes

Recent developments in web standards and web browser technology such as *Web Audio API*, *Web MIDI API* and *WebGL* made the browser as a platform for an audiovisual performance appealing. Exploring the possibilities and limits of these technologies was another intriguing aspect of the decision to put resources into experiments in the web browser. A significant amount of work went into the development of a prototype and the results looked promising at first. Using the webcam input as a terrain (see section 5.2.2) was easy to do. The system load was relatively high, but was tied to the basic setup without the increasing complexity causing a notable difference in CPU load. However, there were soon audible artifacts, clicks and cracks. The biggest problem was that the wave terrain distortion effect did not express stable paths on the terrain, i.e. the navigator point did not settle on stable periodic paths, but rather moved randomly across the terrain. The resulting audio output was mere noise. The suspicion that the noisy and ever-changing webcam image was the only cause of the issue proved to be wrong when the problem remained after exchanging the webcam input with a static, high resolution, blurred image. Besides the plain JavaScript solution, a reimplementation in *Tone.js* (Tone.js 2017) and *p5.js*, the JavaScript port of *Processing*, was surprisingly easy and finished quickly, but there were similar problems. The exchange of sample data between audio processing and graphics rendering processes poses several challenges that could not well be overcome even by using the desktop version of *Processing* (Fry and Reas 2017), neither with its shipped *Sound* library nor in combination with externals such as *Minim* (Di Fede and Mills 2017) or *Beads* (Bown, Crawford, Porter, and Martin 2017).

Soon, resorting to Pure Data proved to be the right choice, especially because the GEM object *scopeXYZ~* takes care of transferring audio sample data from the audio process to OpenGL. Another important advantage of Pd over other tools I experimented with is the performance which is much better than with the previous technologies. The CPU load was relatively low, even after I had added all extensions to the original, basic processing based on *Asterism* (section 3.2.3) for the final performance. This is remarkable considering that the JavaScript solution challenged the CPU at a sample rate of 44.1kHz while Pd worked smoothly at 96kHz. However, before the first experiments with 96kHz, the architecture had already been adjusted to exploit multiple CPU cores (see section 5.3) and therefore the comparison is not particularly fair. Personally I have found that, being a visual programming language, Pd takes away a certain degree of complexity and with it also some of the flexibility compared to textual programming languages. But it enforces a clean program structure and data flow design because the graphical user interface reflects a chaotic setup in a direct and blatant way.

4.2 Pure Data

Pure Data is a visual programming language released by Miller Puckette in 1996 (Puckette 1996a; Puckette 1996b). He first developed Max in the mid-1980s and by writing Pure Data from scratch, he intended to address certain shortcomings of the system such as handling of compound data structures (hence the name) or support for non-audio signals like audio spectra, graphics and video. Max's initial purpose was the creation of a real-time audio system that is usable by artists and other people who do not necessarily have a background in computer science and computer programming. It did so by imitating the behavior of patchable analog synthesizers by means of a clear, simple graphical user interface whilst maintaining a good real-time control strategy. This was a compromise between two seemingly divergent priorities that had not been targeted well by previous solutions. Soon Max was used as a programming language which it had not originally been destined for and thus certain shortcomings surfaced, especially with respect to efficient use of data. For this reason a new system named *Animal* was developed at IRCAM. Many of Pure Data's innovations have their roots in *Animal* (Puckette 1996b, 1).

Pd distinguishes between audio (sample) data and messages. Messages can carry different kinds of data such as floating point numbers, strings, lists and so called *bang*s, which are used to trigger various operations. Most programming operations are performed using objects, messages, numbers, arrays and tables. Additional building blocks are available, but they only provide convenience functionality. Objects may have any number of audio and/or message inputs and outputs. The leftmost input generally triggers processing of the respective object, i.e. changes on additional inputs do not affect potential outputs until the first input is triggered. Inputs and outputs of different objects are connected by dragging lines between them, using the mouse cursor.

There are three different methods to structure patches and extend their functionality. *Subpatches* appear as single objects in one patch and merely wrap parts of the main patch which can then be edited in another window. *Abstractions* are external Pd patches stored in separate files that can be reused multiple times in one or more patches as new objects. Unlike with subpatches, later changes in abstractions affect each instantiation whereas duplicates of a subpatch are not automatically updated along with the original. The last way to add building blocks to Pure Data are *externals* which are objects that wrap compiled computer programs, usually developed in C or other programming languages that support the programming interface.

The graphical front-end uses the portable *tk* toolkit. Pd's architecture design was focused on uniprocessors (Puckette 1996b, 3) which likely is the reason why current versions are still not capable of exploiting multicore processors which have been the standard in personal computing for several years. This is especially unfortunate for audiovisual projects, where a separation of audio and video processing into different execution threads would allow them to run on different CPU cores and thus push the limits of feasible computational complexity of Pd patches. By splitting up such patches, running them in separate Pd instances and letting them communicate via local network sockets or an audio server capable of routing audio signals between applications, this problem can be circumvented.

Pd is released under an open source license and distributed at no cost, in contrast to Max/MSP, which is a proprietary product. The Pd version employed within this project was 0.47.1 vanilla.

4.3 GEM

The *G*raphics *E*nvironment for *M*ultimedia was originally designed for use in Max, but soon the efforts on Pd solved some issues related to data handling that made it the natural environment for GEM to be embedded in (Danks 1997). It is based on OpenGL and therefore offers non-programmers an interface to 2D and 3D graphics and video, sharing a similar objective as Pd does for the audio DSP domain. It is seamlessly integrated into Pd like any other library and provides a number of objects that are used alongside other Pd objects. All objects that represent operations, transformations and settings in a rendering chain are connected in sequence, passing along the so called *gemlist*.

Chapter 5

Eden

Eden is the name of the prototype that was developed along with this thesis. It can be split into three main parts:

Aaron A synthesizer that generates the input signal.

Wave Terrain Distortion An audio signal processor inspired by wave terrain synthesis which causes a kind of harmonic distortion and is the essential mechanism that generates the y-axis component for an XY-oscilloscope and thus one of the most important factors that sets the system apart from other kinds of oscilloscope art.

Visualization This is where the color dimension comes into play, four audio channels are mapped to two overlayed oscilloscope outputs and stereo divergence is translated into an increased distance between the two oscilloscope layers.

5.1 Aaron

The considerations that went into *Aaron* were mostly of a creative nature, meaning that although implementation work on the different parts of the project was intertwined, the synthesizer was the last development focus because it explores the artistic possibilities enabled by the audio processor and visualization components. A subpatch contains a number of GUI elements to change several parameters (fig. 6.2 on page 33). The same elements can be controlled by MIDI signals.

My MIDI controller is a *Novation Launchkey™ Mini MK2* (fig. 5.1) which has a very limited number of keys and other controls. In order to be able to control a higher number of parameters, I developed a system that allows me to assign all eight knobs to different functions depending on which of the pads is currently pushed.

Most of *Aaron*'s features are technically rather simple. The following paragraphs shall provide an overview of the different parts. We will discuss their creative use and parameters in chapter 6.

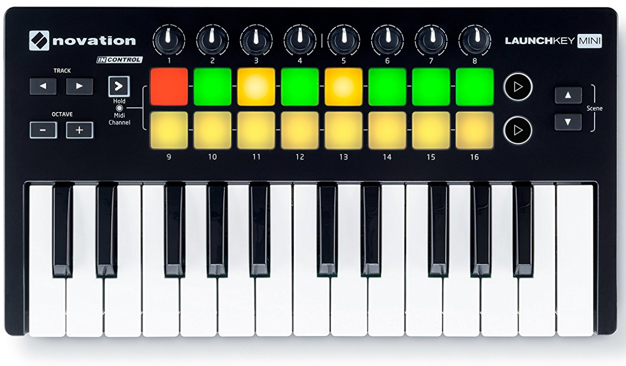

Figure 5.1: Novation Launchkey™ Mini MK2[1]

`fador~` One basic building block that was reused in several locations is the `fador~` abstraction. It is an oscillator with one inlet for the frequency and a second one that allows fading between cosine, triangle and sawtooth wave shapes (Pd patch in fig. A.1, appendix A).

`balance~` Most modules within *Aaron* make use of an abstraction called `balance~` that splits the mono signal arriving at its first input into two separate channels, fading between them based on the coefficient x on the second input which is a floating point value between 0 and 1. The utilized pan law takes the square root of the coefficient and its inverse $(1-x)$, respectively, to determine the factors to multiply the audio stream with for either channel. This way the visual response reflected by shrinking and growing shapes feels more smooth and natural. With the linear solution which omits the square root, sizes increase too little before and too much past the 0.5 center position for my subjective judgment. Fig. A.2 in appendix A shows the according Pd patch.

`mtof_just` Pd provides an internal object named `mtof` that converts a MIDI note index into a frequency based on equal temperament tuning with the common concert A at 440Hz corresponding to the index 69. The translation is given by the formula $f = 440 \times 2^{(m-69)/12}$ where m is the note index. For reasons illustrated in section 2.2, a just tuning system was desirable to achieve steady figures in our visualization. `mtof_just` is an abstraction that accomplishes this by using a 7-limit just tuning (see section 2.2 on just tunings). The reference note for the tuning is set by a creation argument as well as a second inlet which determines the MIDI note

[1]source: https://www.amazon.com/Novation-Launchkey-Keyboard-Controller-LAUNCHKEY-MINI-MK2/dp/B00VVNOMGI

index. Internally, the corresponding frequency is calculated by the standard `mtof` object. Other frequencies then follow from discovering the number of octaves from the reference note and by multiplying with a specific factor that depends on the respective interval (see table 2.4 on page 11). This abstraction is used throughout *Aaron* for note index to frequency translation. See fig. A.3, appendix A for an image of the Pd patch.

Polyphonic Synthesizer The first major component is a polyphonic synthesizer with four voices. Each one consists of a `fador~` and a sine oscillator which may or may not be mixed in to create beat frequencies whose frequency ratio and gain can be adjusted by controls. Voices make use of the envelope `adsr` abstraction from the Pd example library with a 50ms attack and 150ms release and a sustain at 100%, to simply start and stop playing a note according to the keyboard actions and to avoid clipping artifacts.

DC offset Low Frequency Oscillator While not too relevant acoustically, adding the output of a `fador~` based low frequency oscillator to the mix enables us to move the oscilloscope output across the screen horizontally. As we will understand from section 5.2.1 this affects the harmonic spectrum of the distortion effect, the navigator point's elevation and consequently the visuals' y-positions as well. Three control elements allow us to determine frequency, waveform and "expression". The latter combines gain (i.e. amount) and a measure for how right and left channels correlate, i.e. if the offset is applied in the same or opposite directions.

Amplitude Modulated Noise Another LFO with a frequency of up to 10Hz is multiplied with a white noise source. Frequency, amount and stereo balance can be set by the user. The output signal is multiplied with the root mean square of the polyphonic synthesizer module's output signal such that the amount of noise added to the mix depends on the synthesizer volume. We will take a look at its effect on the visuals in section 6.2.4.

Drums A basic drum patch is comprised of three synthetic layers: kick drum, snare and hi-hat. Overall and individual instrument gain levels as well as combined stereo balance are controlled by the user. See fig. A.4, appendix A with the patch for details.

Bass Line Another component outputs a static bass line in A minor that uses just intonation and is eight bars long. The source combines a distorted sawtooth wave and a cosine oscillator at synchronized frequencies read from a static array. The signal then passes through a low-pass filter which is modulated by yet another LFO whose frequency is tied to an integer ratio of the global tempo in beats per minute. It is adjustable along with the overall gain, stereo balance and the filter's center cutoff frequency between 80 and 500Hz. This an exclusively musical choice because the effect of the exact values for the cutoff frequency on the visuals is negligible. You can find the patch in fig. A.5, appendix A.

High Frequency Stepped Wave Form Fenderson published a video tutorial on YouTube (Fenderson 2014), where he explains in detail how he draws mushrooms on an oscilloscope. In this video he also mentions a trick that allows him to duplicate whatever is displayed: By adding a high frequency square wave to the oscilloscope's input, half of the time the samples appear in one location, the other half at an offset according to the square wave's amplitude. If the frequency is high enough, we can see the same display twice. Inspired by this technique I added a feature that separates the output signal into n instances where $n \in \mathbb{N}$. In order to achieve this, samples of a sawtooth wave f_{saw} are reused as in $f(t,n) = \frac{\lfloor f_{saw}(t) \times n \rfloor}{n}$. The result is a stepped function with n steps. The distance between instances can then be adjusted by changing a factor s that is multiplied with $f(t,n)$. The implementation (fig. A.6, appendix A) allows us to vary n (between 1 and 8) and s. The frequency interval is centered around the sample rate and can be adjusted by ±1000Hz.

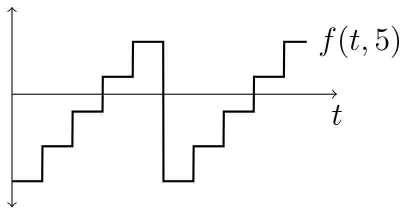

Figure 5.2: Stepped wave form.

5.2 Mechanism

The basic visualization process is that of two oscilloscopes in XY-mode. Since we are using a digital system and only draw disconnected points for every pair of input samples, the use of an oscilloscope can hardly account for any kind of interesting shapes and structures. The heart of the translation and the main reason why shapes and figures emerge is the audio synthesis and signal processing that precedes the visualization. From watching the video of *Asterism* (Cooke 2016) as well as *Eden*, it is easy to tell that the two "characters" use the same algorithm which is only duplicated. In order to allow for symmetric figures, the left channel's image is mirrored horizontally. For both instances, the two input channels are defined by the audio signal *before* (x-axis) and *after* the signal processor (y-axis). Let us now take a look at the underlying concept of the signal processor.

5.2.1 Wave Terrain Synthesis

Wave terrain synthesis is a synthesis method where an audio wave is generated by navigating a point across a three-dimensional terrain along a periodic path and sampling the elevation at a specific XY-position for each sample. The frequency is determined by the length of one period in samples or, in other words, the speed at which the path is navigated and repeated. The wave shape and in consequence the timbre can be varied continuously by modifying the path. Normally, the path will be an orbit so that there are no jumps in amplitude and thus very high harmonic frequencies, given a continuous terrain with no harsh edges. Also, the terrain height is usually normalized to values in the interval $[-1; 1]$ corresponding to the range of digital audio sample values to avoid clipping. In mathematical terms, the terrain lookup function $f(x, y)$ is a mapping from two-dimensional to one-dimensional space (Borgonovo and Haus 1986).

f can be an arithmetic function, where the terrain is defined continuously and thus the elevation is defined and can be calculated for every possible $x, y \in \mathbb{R}$ in a given interval on both axes. It is also possible for f to be a lookup function on a discrete matrix. Values between the defined elevations, i.e. where $x \notin \mathbb{Z}$ and/or $y \notin \mathbb{Z}$, we need to perform an interpolation function. Depending on the complexity of the function and interpolation method, a table lookup is usually computationally cheaper than an arithmetic calculation and therefore, discrete lookup functions are generally better suited for real-time synthesis. While the audiovisual performance does not use wave terrain synthesis in this basic form, the concept is very similar.

5.2.2 Wave Terrain Distortion

Based on wave terrain synthesis we lookup elevation values on a terrain, but instead of feeding two independent signals into either input, we take one audio signal as input to the x-axis and previous output values at a variable delay for the y-coordinate. If the terrain is reasonably smooth and the signal harmonic and simple, this not only results in a periodic path, but it also increases the effect of the terrain's character on the signal because it is fed into the system twice. The length of the delay line determines how long it takes for a changing signal to establish a feedback loop and thus a stable harmonic translation. See fig. 5.3 for a schematic overview of the dataflow of a process that I suggest we refer to as *wave terrain distortion*, which can also be viewed as a kind of wave shaping distortion.

In section 3.2.3 we saw that Cooke used Max and Jitter for *Asterism*. Jitter of-

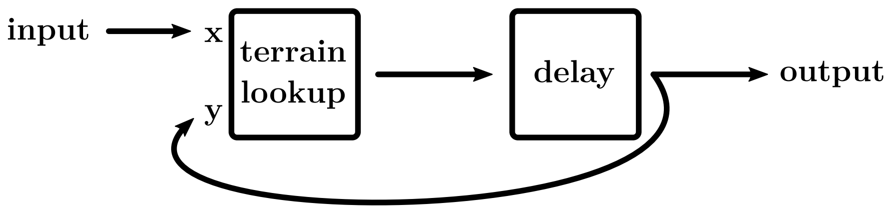

Figure 5.3: data flow in the modified wave terrain lookup

fers a NURBS (*Non-uniform rational basis spline*) functionality which is capable of generating a 3D surface on the basis of an input matrix with discrete data points by using B-spline curves and thus a rather smooth plane from input with a limited resolution. The input terrain is in fact a low resolution (100x100px), blurred grayscale image. The image's gray values are loaded into a matrix which is then passed to the NURBS module. The edges of the terrain are all at zero elevation to allow for wrapping (Jethro Cooke, pers. comm.).

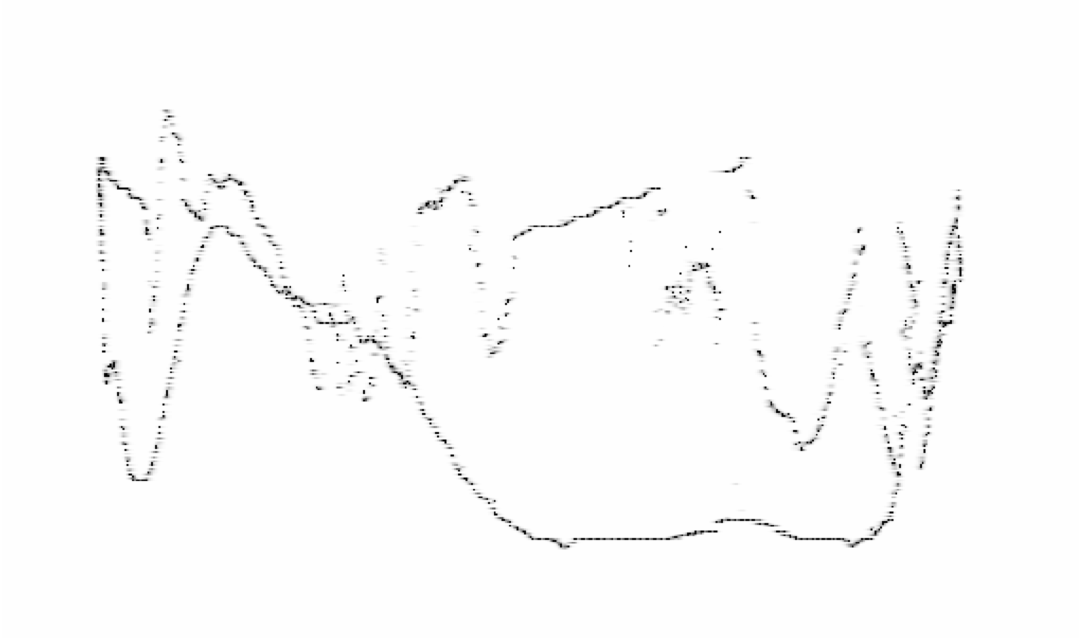

Figure 5.4: blurred 8-bit grayscale image terrain with sine wave input

The initial plan for this project was the use of the webcam image to generate the terrain. However, it turned out that the image was too noisy and just not smooth enough for use as a terrain. Subsequent experiments with still, heavily blurred images still led to obvious artifacts caused by the limited bit depth of a grayscale image (see fig. 5.4). The displayed paths were not stable and expressed a lot of noise, i.e. changing outliers. Pixels appeared at stepped positions on the y-axis because GEM (section 4.3) has a color resolution of 8 bits per channel. This means that we have a maximum of 256 different positions on the y-axis. The solution was to use a two-dimensional arithmetic function instead of an image. Pd keeps numbers as

32-bit floating point values which gives us a resolution of $2^{32} \approx 4.3 \times 10^9$ values. This certainly serves our purpose. Matrix lookups with interpolation are usually not as computationally expensive as arithmetic calculations. However, the cycles added by using the arithmetic function introduced in the following section did not pose a noticeable strain on the performance.

A suitable arithmetic function needs to fulfill a number of criteria. First of all it needs to be two-dimensional because we are looking at a terrain with x and y-coordinates. If the function's output is a simple plane with little changes in altitude, i.e. no bumps and low-points, there will be hardly any variations as the navigator point moves across the terrain resulting in horizontal lines or long, horizontal ellipses and either no audible output or a monotonous harmonic spectrum. And even in the case of pronounced waves and bumps, the result will express little variation as the lookup parameters change if the terrain has too much regularities. On the other hand a rough terrain may lead to a harsh harmonic spectrum and thus result in a acoustically and visually "dirty", unpleasant output. In summary we can say that a desirable surface would have predictable, mellow irregularities so that it is possible to anticipate what certain changes in the input signal may do to the output, whilst still allowing for a number variations.

Many of these properties do not so much depend on the used type of function but rather on the selected parameters. We can scale every function on every axis. However, some resulting planes may be to simple or even.

Schwefel's Function No. 2.26 (Schwefel 1981) was chosen because it produced the artistically most usable results with respect to the above criteria, from a number of different two-dimensional functions that were tested. It is defined for any number of dimensions. For our purposes, the definition for two dimensions is given by

$$f(x,y) = -x\,sin\left(\sqrt{|x|}\right) - y\,sin\left(\sqrt{|y|}\right). \tag{5.1}$$

Some experimenting yielded predictable and not too distorted results for $x = 10\,x_0$ and $y = 10\,x_0$ which expands the lookup window for a more versatile output. In order to normalize the audio output, it is then divided by eight.

Fig. 5.5 shows the terrain with all these parameters taken into account. Fig. A.7 in appendix A contains the Pd patch which calculates Schwefel's Function No. 2.26 with the selected parameters.

All-Pass Filter

In *Asterism* (section 3.2.3), Cooke (Cooke, pers. comm.) occasionally activates an all-pass filter after the wave terrain distortion effect. As this changes phase relationships between different frequencies, it also influences the navigator point's path and

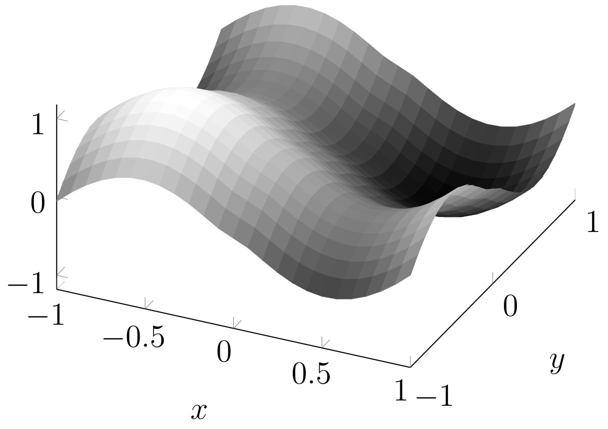

Figure 5.5: 3D plot of scaled Schwefel's function no. 2.26

thus the image. Since the application of distortion adds overtones, it makes sense to include this filter after the signal processor and not on the initial input. The implementation of the filter uses two primitive filters, a non-recirculating (`rzero_rev~`) and a recirculating one (`rpole~`). It was copied from the Pd patch library and is documented in (Puckette 2006).

5.2.3 Visualization

We have already seen that the visualization displays a sequence of pixels where x is the current input sample *before* the wave terrain distortion and y the last processed sample from the delay line.

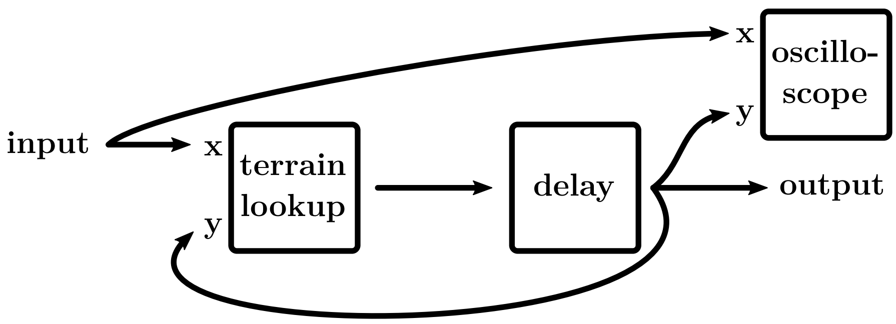

Figure 5.6: data flow with XY-oscilloscope

Fig. 5.6 extends fig. 5.3 from section 5.2.2 with the oscilloscope. Note that the input parameters to the terrain lookup are the same as the oscilloscope's input. What this means is that what we are seeing is the visualization of the navigator point's path across the terrain from a bird's eye perspective. The oscilloscope is set to project the last 4500 input sample pairs for every rendering cycle.

5.2.4 Afterglow

Having worked on the Pd patch for a while, I decided that the image was to "clean" and should feel more dynamic and organic. The motion blur effect provided by GEM was not strong enough to satisfy my ideas, so I found a way of accomplishing what I desired. The afterglow effect I was looking for was implemented by rendering the original output to a frame buffer and alpha blending it with a slightly darkened version of the previous frame.

Coincidentally I discovered that an interesting depth perception can be produced when the previous frame is not only darkened but also carefully scaled, i.e. either grows or shrinks slightly. A later addition was a variation of the scaling factor that depends on the joined stereo output's amplitude, to make things even more organic.

5.2.5 Color Mappings

Early in the project, the idea of adding color to the visuals came up. The final solution certainly leaves room for improvements, especially with respect to chromesthetically informed mappings. As we saw in section 2.1.1, there seems to be a tendency to associate low frequencies with "dark" colors such as blue and violet. Higher frequencies are usually connected to "lighter" colors like yellow and green and midranges with red and orange. The translation therefore merely splits the frequency spectrum into three bands, continuously takes their respective root mean square values and maps them to the RGB values, where 30-250Hz is the bass and determines the blue channel, 250-2000Hz corresponds to the mids and the red channel and everything above 2000Hz is mapped to green. Frequencies lower than 30Hz are ignored to prevent low frequency oscillators from affecting the color. This is reasonable because infrasound is not audible as a steady pitch and thus we can assume that it will not contribute to perceived congruence when mapped to dark, blue colors, which is the intention for the mapping. Color values can only change slowly and therefore constantly fade to new values unless the RMS stays the same for more than 150ms. A potential problem with using the RMS is that it does not always reflect perceived loudness, especially when there is a DC offset in the signal. As a subjective choice, I worked with different mappings during development that felt like a relation of pitch and "warmth" (red - low range, green - midrange, blue - high range). As a more scientifically informed option I later switched to the mappings explained above. Even though I had gotten used to a different behavior, I found the results surprisingly pleasing and the perceived coherence, in my opinion, works quite well. Fig. A.8 in appendix A shows the implementation of the translation of sound into RGB values.

5.2.6 Stereo Divergence

Another addition to the dynamics of the visualization is inclusion of the stereo divergence. This idea has its roots in an initial plan to incorporate narrative elements. The visualizations of both stereo channels drift further apart when the volume of the side channel (i.e. difference between left and right) increases. This factor also depends on the individual amplitudes. Please refer to the Pd patch for the details (fig. A.9, appendix A).

5.3 Architecture

In section 4.2 we learned that Pure Data was specifically designed to run efficiently on a single uniprocessor. Even though the process of *wave terrain distortion* is closely coupled with the visuals, it is possible to split audio processing and OpenGL operations into separate Pure Data patches. This was then performed later in the process, in order to allow for more resource intensive operations in the future and to make recording and reusing each stream separately easier, potentially with additional software. Now it was possible to run audio and visual patches in two different instances of Pd simultaneously. As a consequence, the operating system is able to dispatch their calculations to different CPU cores and it enables us to use the JACK audio server to route signals between the Pd instances and other applications. Fig. 5.7 lists the basic routing setup. Output channels 0 and 2 of the audio instance are the synthesized input signals that are also internally sent through the wave terrain distortion processor to produce the final audio output available at channels 1 and 3.

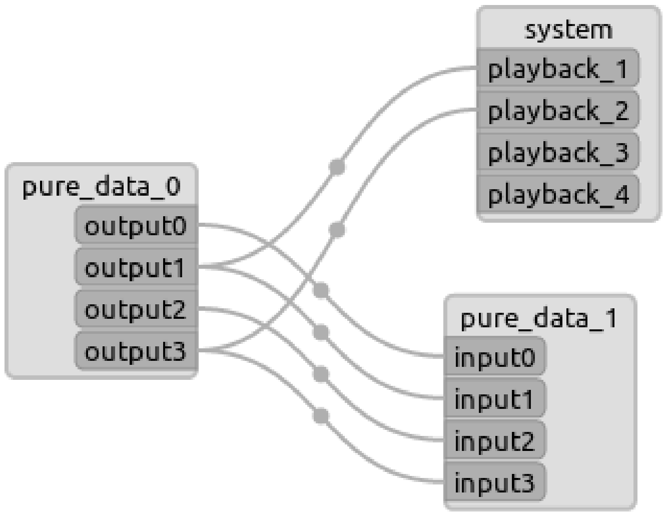

Figure 5.7: JACK routing setup between Pd audio (*pure_data_0*) and video (*pure_data_1*) instances

Chapter 6

Artistic Application

In this chapter we will take a look at methods and features of the synthesized audio signal that is fed into the wave terrain distortion processor (section 5.2.2) and how they create and affect the visual output. In order to demonstrate the mentioned relationships and showcase an artistic sample application of *Eden*, a video was produced and made available along with this document that points out the problems and properties discussed here. In addition to explaining audiovisual relationships, we shall critically reflect congruencies and divergencies and their contribution to the quality of the overall experience in reference to the declared objectives for this project (see section 1.2).

The visualization generally consists of a black canvas and colored plots. However, many of the screenshots in this chapter will benefit from inverted grayscale images because in print, the features they serve to demonstrate may be more pronounced this way.

6.1 Sample Rate

During early stages of the development, the system worked at a sample rate of 44.1kHz. But long, periodic paths at high frequencies appeared as a series of disconnected points, rather than one closed orbit which is desirable because as a consequence the overall image is more detailed and thus allows for more nuances to prevent a monotonous image. Increasing the number of displayed samples per frame was a possible candidate for optimization, but the selected value of 2000 had already been chosen as a compromise between temporal and visual resolution: A higher number of samples led to the visuals behaving stagnantly, persisting for too long and thereby, in my opinion, reducing the immediacy of the translation from sound to image. The obvious resolution therefore was switching to 96kHz, which is the maximum sample rate supported by my setup. In order to maintain the same degree of volatility, the oscilloscope displays samples from the same timeframe when the number of plotted samples per frame is increased by the same ratio as the sample rate. The resulting value of 4354 was rounded up to 4500. Fig. 6.1 compares a section of the same sine frequency for both settings.

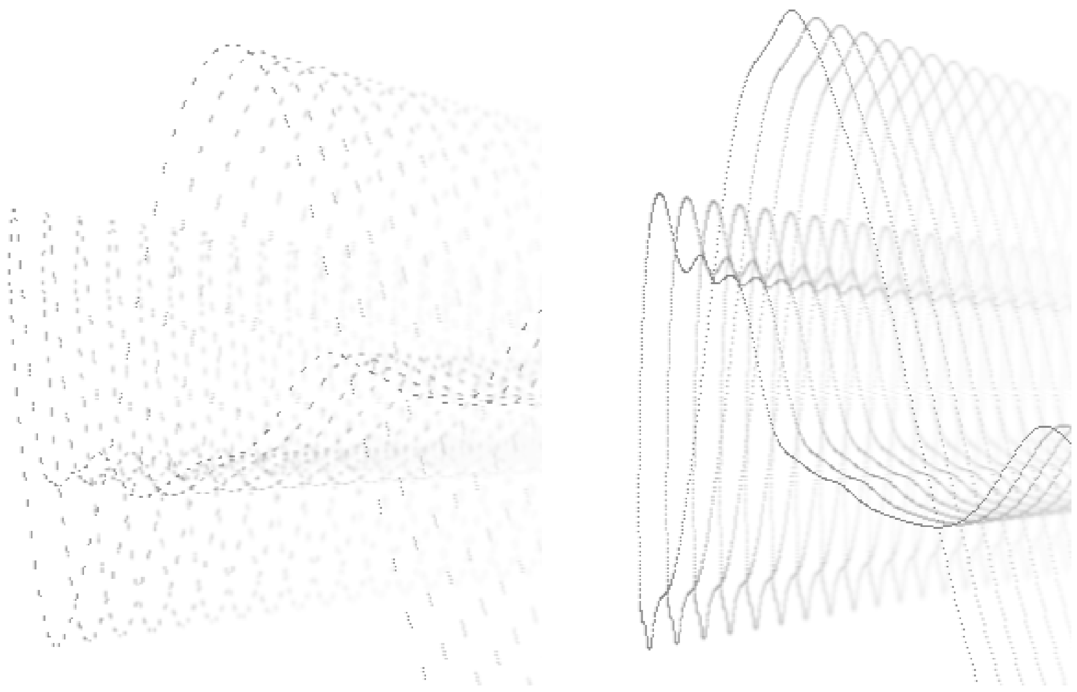
Figure 6.1: Comparison of 2000 samples at 44.1kHz (left) and 4500 samples at 96kHz (right).

6.2 Controls

In *Aaron* (section 5.1) there is a subpatch that centralizes all the controls. Fig. 6.2 shows this patch and in this chapter we will go through *Aaron*'s features which are all represented by elements in this figure. Numbers relate to the mappings with the MIDI controller (fig. 5.1). We will not discuss these because they are irrelevant for understanding the outcome.

6.2.1 Common Parameters

The following three parameters concern the wave terrain distortion signal processor and therefore affect the final output which includes all *Aaron* modules.

Terrain Height

The terrain height refers to a factor that is multiplied with the lookup elevation values. Increasing this factor translates directly to a vertically more expanded orbit

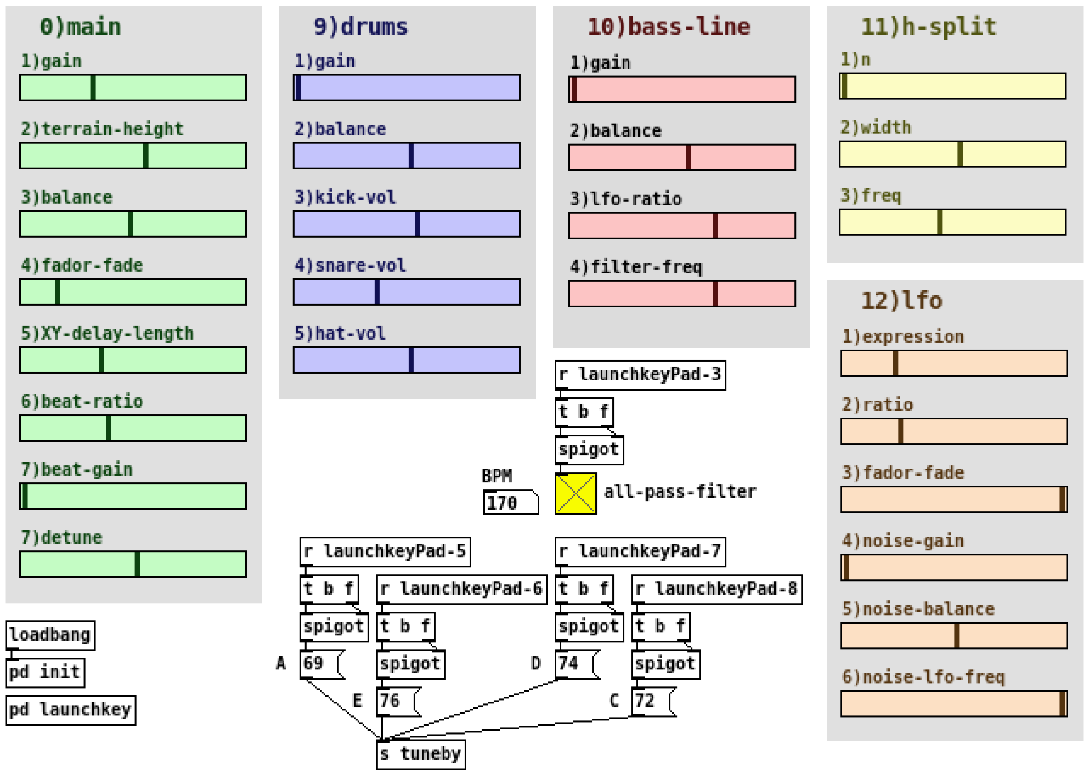

Figure 6.2: *Aaron* `controls` subpatch.

and thus an increased output amplitude. It therefore determines the character of the applied distortion and output gain simultaneously. Note that the specific terrain's elevation shows a tendency to get more pronounced with increasing distance from the center at (0|0) and therefore an increased expansion in the navigation point's orbit will usually cause a more pronounced distortion effect.

XY Delay Length

Section 5.2.2 explained how the terrain lookup function accepts its own output at a variable delay as its y-input. If there was no delay, a changing input frequency would not change the visual shape. The visual reflection of the delay length relates to the input frequency because it changes the matching of x and y values, i.e. the phase offset between the two signals. With respect to an actual performance, I have found the exact length of the delay line to be of little relevance, as long as it is greater than about ten milliseconds to avoid consecutive notes looking too similar. With this prerequisite satisfied, the exact shapes for any given frequency feel very unpredictable, even in lower ranges where frequencies do not differ as much as in higher ranges and thereby a desirable level of variety is maintained.

All-Pass Filter

The all-pass filter as introduced in section 5.2.2 can only be on or off. While there is a fading effect between the two states, the controls do not support continuous values. The differences between the states are too subtle to make continuous controls useful in this context. Also, there are so many other parameters in the system that reducing the number of possible combinations was somewhat desirable from an artistic point of view. With low frequencies, the effect is unnoticeable. In higher ranges, the changes to the visuals are similar to a changing delay length: The exact shapes are morphed in an unpredictable way, but the overall character, i.e. smoothness and roughness, remains the same. Unlike with a modified delay lenght, the morphing motion is vertical, rather than horizontal. This behavior ensures a desirably surprising element. With noisy sounds and digital artifacts caused by frequencies above the sampling rate, the visuals tend to become more chaotic because single pixels seem to become displaced (fig. 6.3). These observations are purely subjective and discussing the technical reasons and properties of the changes would require a mathematical investigation of digital artifacts, all-pass filters and the type of implementation at hand. This shall not be covered within the context of this thesis.

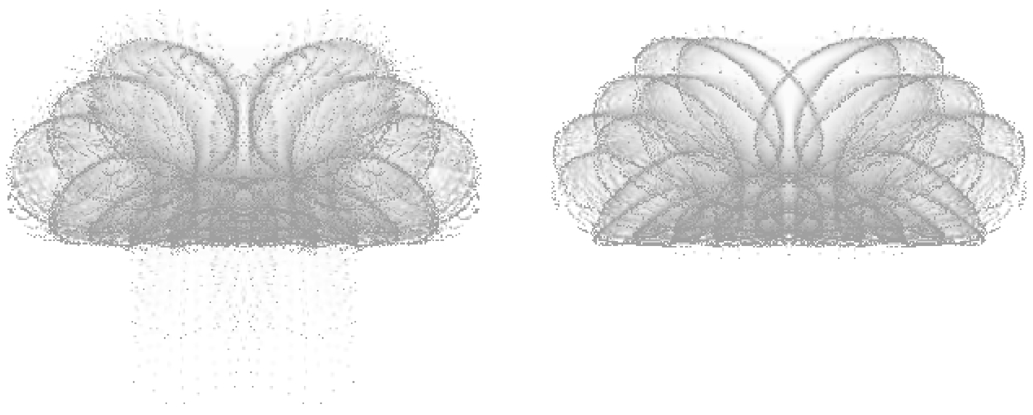

Figure 6.3: Effect of an all-pass filter on frequencies above the sampling rate (effect left, reference right).

6.2.2 Polyphonic Synthesizer

A brief technical introduction to the polyphonic synthesizer was given on page 23 in section 5.1. Generally, the *gain* of an input signal defines the horizontal expansion of the navigator point's orbit. Similar to vertical expansion, the audio output not only expresses a more complex harmonic spectrum but the overall gain behaves roughly proportional to the input for reasons explained in the previous section.

The `fador~` object was already explained in section 5.1. As one would expect from the translation mechanism, cosine waves create very round, smooth shapes. Fading to a triangle wave causes the expression of corners and a saw tooth further produces disconnected parts of a figure (see fig. 6.4).

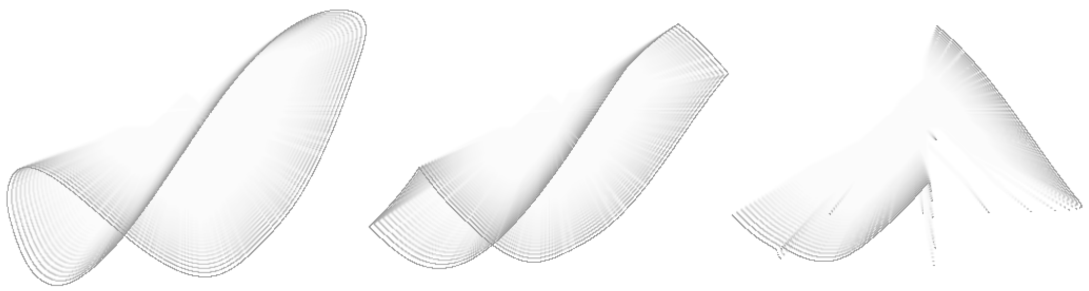

Figure 6.4: Example visuals for simple waveforms. From left to right: cosine, triangle and sawtooth (at equal frequency and amplitude).

Live *detuning* of a note morphs its representation's shape horizontally. The behavior is the same as, but more subtle than the consequences from changing the length of the delay as mentioned in the previous section, i.e. the motions result from a change in the phase relationships between x and y-components of the terrain coordinates.

Beatings

Mixing in a cosine oscillator at a similar frequency as the voice's main frequency creates beatings. Depending on the amount and thus the volume of the beats and their frequency, shapes wobble, apparently rotate and become more entangled. At higher main frequencies, sometimes outlines become thicker and express various kinds of patterns. Figure 6.5 shows an example.

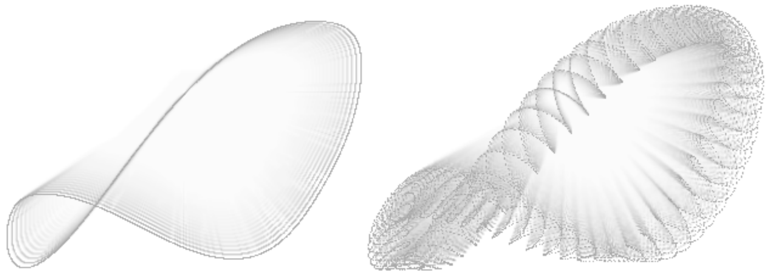

Figure 6.5: Example figure with (right) and without (left) beat frequency.

6.2.3 DC Offset Low Frequency Oscillator

Adding the amplitude of a low frequency oscillator to the mix moves the navigator point across the terrain horizontally. While this does change the terrain lookup coordinates, the effect is mostly visual. We already know that there are two parallel wave terrain distortion processes happening in *Eden* for the two stereo channels. The visual representation of the left terrain lookup is mirrored. If both channels emit the same signal, the overlaid visuals result in a symmetric image. This is relevant for understanding the effect created and modulated by the *expression* parameter: The center position equals to no DC offset and no consequences on the visuals. Moving the fader position to the right adds an increasing amount of the LFO amplitude equally to both channels. Visually, this means that the navigator points slowly oscillate symmetrically away from and towards each other around the center of the output window. The effect of the fader's left half adds the LFO amplitude to one channel, while subtracting it from the other. The visual result is that the navigator points still oscillate, but this time their directions are synchronized. From this follows that the terrain lookup positions are now different between the two channels and as a result, the stereo divergence (section 5.2.6) takes effect and the movements are not strictly synchronous anymore. The output is a very organic interplay between the points that I would like to compare to the movements of two fish swimming in water, alternately drifting apart and then returning to the other one. These motions are very interesting in that they are not obviously related to audible properties, therefore not predictable and seem to add a refreshing randomness to the image that complements the more obvious congruencies that may be easy to anticipate.

Two other controls serve for setting the LFO *frequency* between 0 and 1Hz and switching the `fador~` *waveform*. A low-pass filter at 7Hz causes a briefly visible transition and thus a visual effect rather than a plain jump when using a sawtooth wave.

6.2.4 Amplitude Modulated Noise

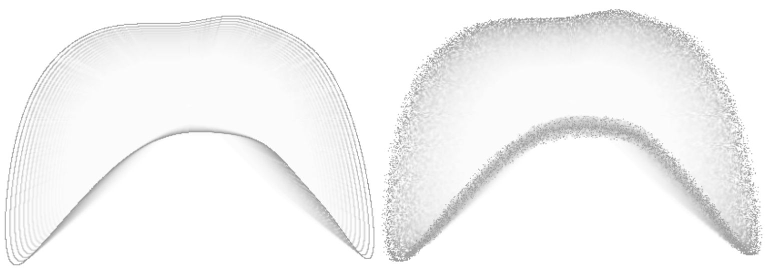

Figure 6.6: A sine wave (left) with noise added (right).

The addition of noise offsets the lookup coordinates by random amounts for each sample (see fig. 6.6 for an example). Another LFO amplitude modulates the amount of noise that is added. The gain is also dependent on the RMS of the polyphonic synthesizer's output so that the amount of noise increases with its volume and is not present when no note is being played. Noise *gain*, *balance* and LFO *frequency* ($0 \leq f \leq 10$) can be controlled by GUI elements.

6.2.5 High Frequency Stepped Waveform

The concept behind the stepped waveform was already discussed on page 24 (section 5.1). For practical usage it suffices to point out that this mechanism splits the navigator point's orbit into disconnected copies. A possible result is shown in fig. 6.7.

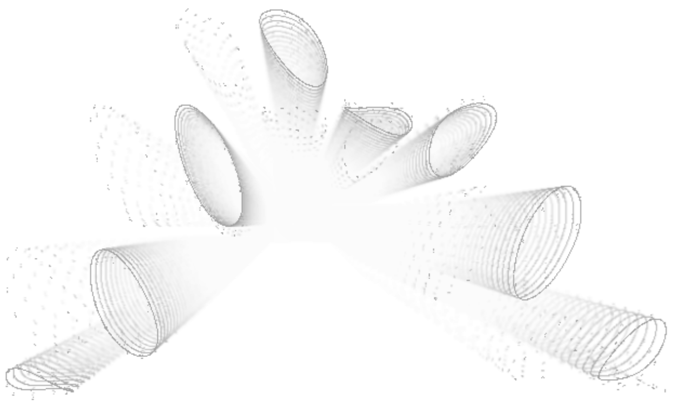

Figure 6.7: Horizontal split caused by high frequency stepped waveform.

While the effect on the visuals might open up a number of new possibilities for combinations with other techniques, the representation in the audio signal is a dominant frequency whose amplitude is tied to the distance between the separate sub-orbits. At a distance that removes overlaps of sub-orbits, the additional frequency is significantly louder than the actual note being played. Therefore usage of this effect is not very versatile if we want to keep the audio part of the performance musically pleasant and maintain variability. The three controllable parameters are the number of sub-orbits n with a maximum of eight, the (equal) *distance* between them, i.e. the stepped wave's amplitude, plus its *frequency* which influences the shapes of sub-orbits.

6.2.6 Drums

A basic drum and bass beat is controlled by a common *gain*, *stereo balance* and the gain values for the three layers: *kick drum*, *snare* and *hi-hat*. Since the kick holds a fixed frequency for a short amount of time, the corresponding visual is a clearly defined figure that appears for a few milliseconds. Snare and hi-hat consist of band-limited white noise with different cut-off frequencies and they cause short, noisy bursts that might be described as clouds whose densities decrease at their outlines (fig. 6.8).

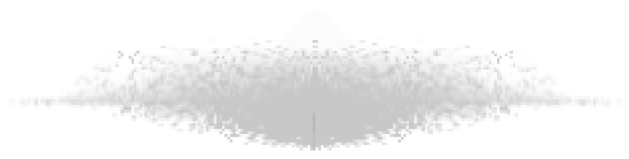

Figure 6.8: Visual image of the hi-hat.

6.2.7 Bass Line

The bass line is a melody I wrote that is repeated infinitely. The MIDI notes' indices are stored in an array and translated to frequencies in 7-limit just intonation (see section 2.2) tuned to an A at 440Hz. The melody would go well with a sequence of four chords, namely A major, E minor, D minor and C major. In order to create stable visual figures the tuning would need to be adjusted throughout the melody to allow for overall just harmonies, should we decide to play chords on top of the bass line.

Controls determine gain, stereo balance plus the cutoff frequencies and rates of an LFO modulated low-pass filter. The rate is tied to the tempo, resulting in a "pumping" sound and wobbling visual whose movement is synchronized to the musical rhythm.

6.2.8 Just Intonation

For experiments with just and unjust intervals, the polyphonic synthesizer can be tuned to the root notes of the four chords listed in the previous section. Fig. 6.9 demonstrates the dramatic difference of the visual representation of the same interval in two tunings.

6.3 Video Production

For the video demonstration, drums and bass line were synchronized to a common clock and ran in an infinite loop. No external software sequencer was used and the audio engine was controlled exclusively with the *Novation Launchkey™ Mini* (fig. 5.1) and the GUI control elements (fig. 6.2). Artistic use of this system is like learning a new instrument and I was not able to create a performance in real-time that incorporates all the above features and satisfies the artistic criteria and objectives I set at the beginning (section 1.2). In order to produce the final video, the

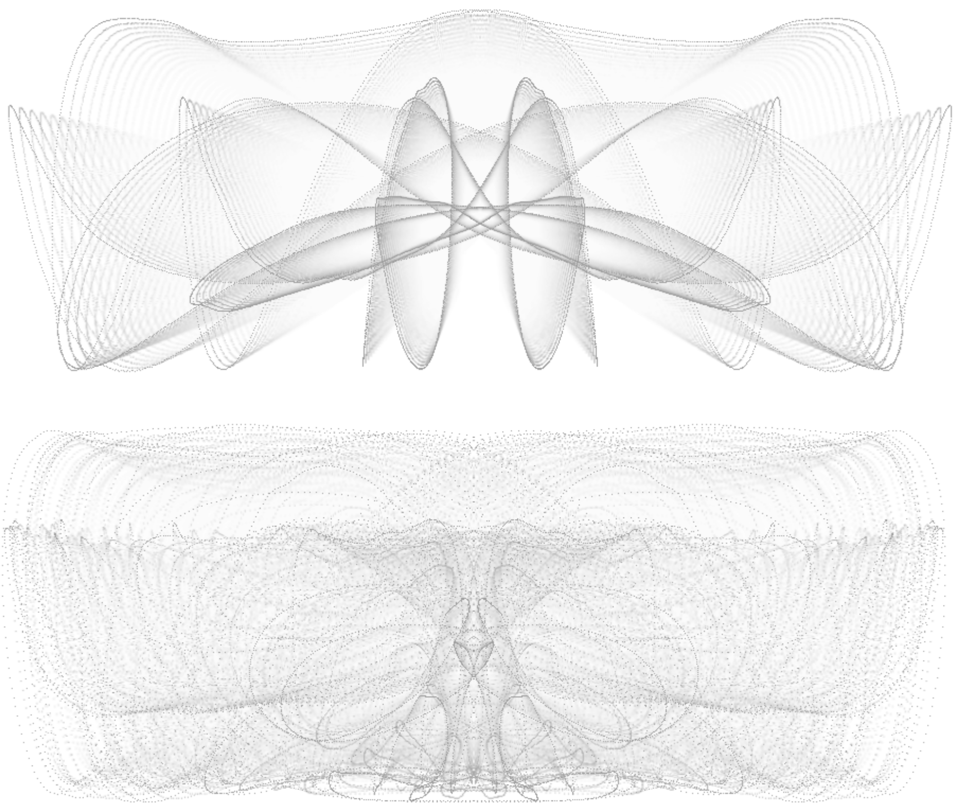
Figure 6.9: just perfect fourth (top) and unjust fourth (bottom).

four channel audio was recorded as a 32-bit floating point WAV file with a sample rate of 96kHz. This file was then imported into an Ardour session and cut to reduce the original material to the most relevant scenes and organize them in a musically meaningful progression, i.e. aviod monotonicity, too many repetitions and abrupt changes of moods and atmospheres. Subsequently, the result was routed to the Pd instance that ran the visualization process and rendered to a video file using the GEM object `pix_record`.

6.4 Discussion

Let us now take a brief look at the final outcome and the relationship of image and sound with respect to what we have learned about the perception of congruence and divergence in section 2.1.

Spatial proximity is not synchronized: With the exception of the stereo divergence translation, left and right channels determine completely independent shapes that can both move across the entire rendering area. There is no stereo panning applied to the audio output that could account for the character's horizontal locations and expansions on the screen. As section 2.1.3 explained, maximizing congruence is not generally a way of improving the artistic and entertaining value of an audiovisual piece. However, the divergence of spatial clues in audio and visual domains is something that in my experience often disrupted the perceived unity of image and sound. This is an aspect that, in my opinion, deserves improvement.

Temporal proximity is exclusively determined by the length of the delay line. Since the implementation puts an upper bound of 50ms on this parameter, temporal proximity in terms of human perception of auditory and visual stimuli is always secured. A longer delay would cause the figures to change very slowly and thus appear sluggish, which is undesirable because it may break the overall impression of coherence if the slow pace is not shared by the audio signal. There might be other ways to introduce variable delays that may add more variation to the presentation. This could be considered as a possible extension because it may improve engagement of the viewers' attention.

Color mappings are an extensively studied area with few secured congruencies that are shared by the majority of people. The current implementation uses a very simple mechanism that I found to work quite well, even though neither the pitch-hue correlations strictly conform to the theory that inspired the implementation nor is the theory a thoroughly tested one. Therefore I believe that the colors create a feeling of congruence because they are obviously and consistently related to basic spectral properties of the audio signal. The exact mappings might not be essential. This hypothesis is consistent with my observation that switching the channels did not impair my subjective experience of the coherence. Section 2.1.1 outlined audio features such as timbre, loudness or major/minor chord harmonies that are more reliably associated to certain colors. During my work on the prototype I suspected that a mapping to timbral features would feel more natural to me than the current solution.

Affective impressions listed in table 2.1 are mostly either inherent to the mechanism. Relationships such as the tempo on both domains, loudness and size or shape and timbre follow from the way the translation from sound to image works. A congruence that could be considered for implementation is that between pitch and vertical position. Currently, these two features are unrelated.

Congruence of idiom is very limited because the expression of distinct shapes occurs only with very simple audio signals. As the visualization is comprised of points, lines and shapes, the stylistic approaches can be said to match for the cases we have considered so far. More complex compositions such as acoustic music or human speech quickly results in noisy, monotonous visuals that hardly reflect the input's idiomatic character.

Atmosphere correlates aspects on a rather high level that will in most cases strongly depend on the composition at hand and how well it aligns with the visualization technique in general.

Chapter 7

Conclusion

In the introduction to this thesis I expressed my artistic vision for this project. The main objective was the implementation and exemplary use of a system that provides audio synthesis mechanisms and a translation of audio data into visuals so that there are immediate connections between the two domains. These connections should enable integrated audiovisual experiences that are more than the sum if their parts. Another declared objective was that the translation balances anticipation and moments of surprise while ensuring that neither auditory nor visual layers necessarily lose a certain degree of aesthetics. Most of these properties are highly subjective. While I set out to realize a personal artistic vision, I would be happy if others copied or reused my system and evaluated subjective impressions and opinions on these points.

The previous chapter discussed a number of inherent congruencies and explained some of the rather volatile and unpredictable properties. The creation of an integrated audiovisual experience that ties sound and image together while being highly engaging is certainly possible with the technology as Cooke has demonstrated in *Asterism*. I believe that my extensions equally enable variability as well as congruence and that it is up to the composer, how engaging and coherent the overall experience can be. My demo video shows some of the new aspects and possibilities. It is a very simple use of the system. More elaborate synthesis systems and experienced users will certainly be able to generate highly interesting compositions that yield very different kinds of visuals on the basis of the same techniques.

7.1 Futurework

Aside from more elaborate audio signals, there is an abundance of possible extensions to the wave terrain distortion and visualization mechanisms. Some of them have been mentioned in previous chapters. One adjustment that the perceived audiovisual coherence could highly benefit from is the translation of spatial location and expansion cues to panning parameters rather than keeping the stereo signals independent. A number of other parameter mappings can be added or modified. For this purpose, more psychoacoustically relevant features such as timbre, tempo and musical properties can be extracted and mapped to visual features such as parameters of the afterglow effect. Particularly, color mappings that more closely reflect scientific findings might be able intensify the audiovisual experience.

In accordance with initial plans, it would be interesting to investigate, how the webcam image may be processed so that it can be used as a basis for the terrain. This would make it possible to build an interactive system where subjects can influence the visuals with gestures or other physical movements.

Furthermore, experiments with shaders provide another area of possible extensions and adjustments to the current visualization algorithm.

Appendix A

Pd Patches

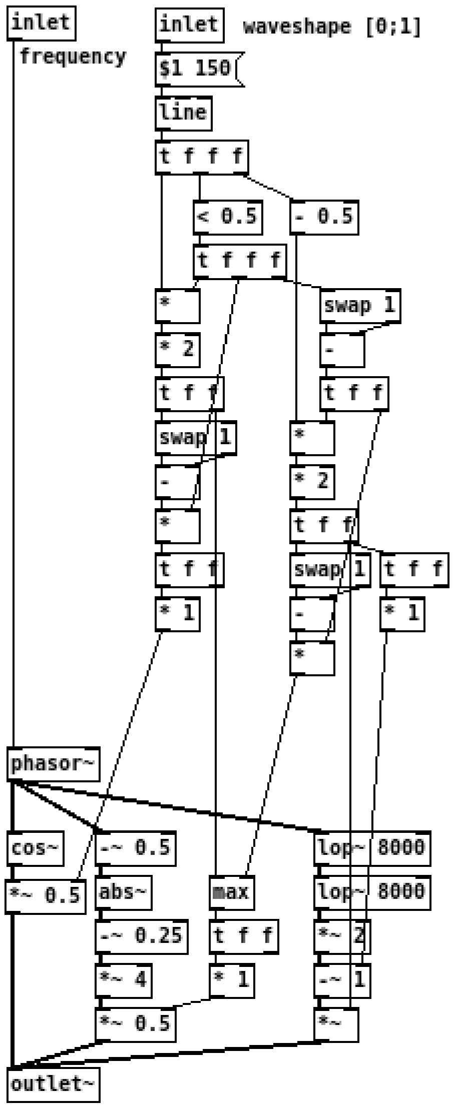

Figure A.1: Pd `fador~` abstraction.

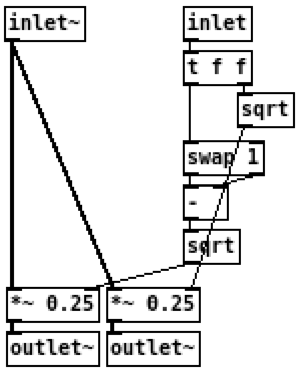

Figure A.2: Pd `balance~` abstraction.

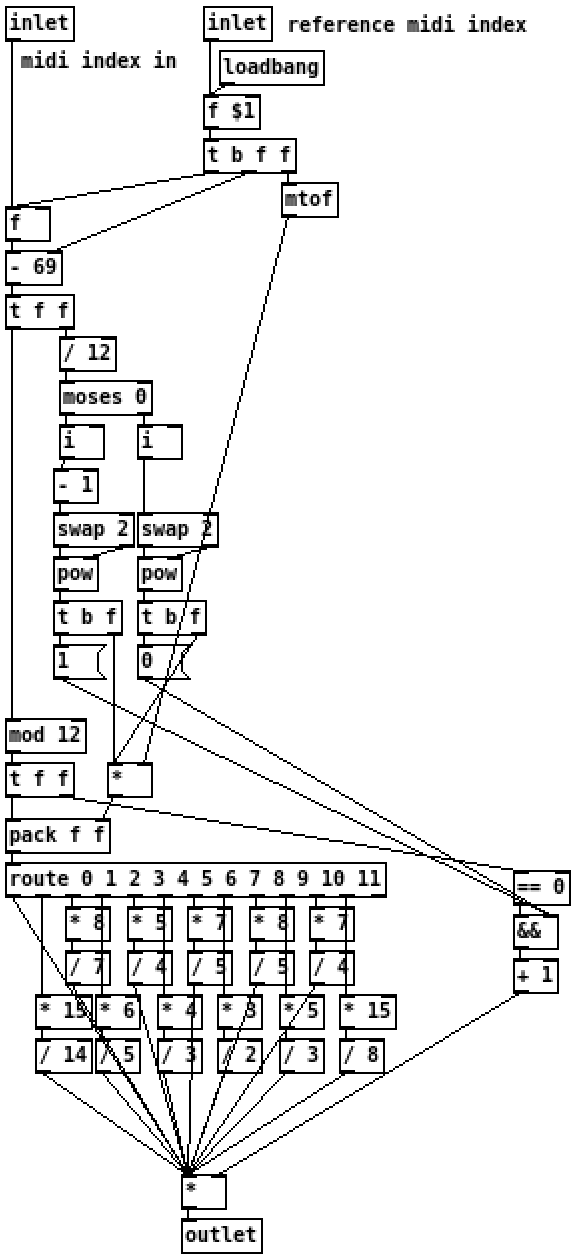

Figure A.3: Pd `mtof_just~` abstraction.

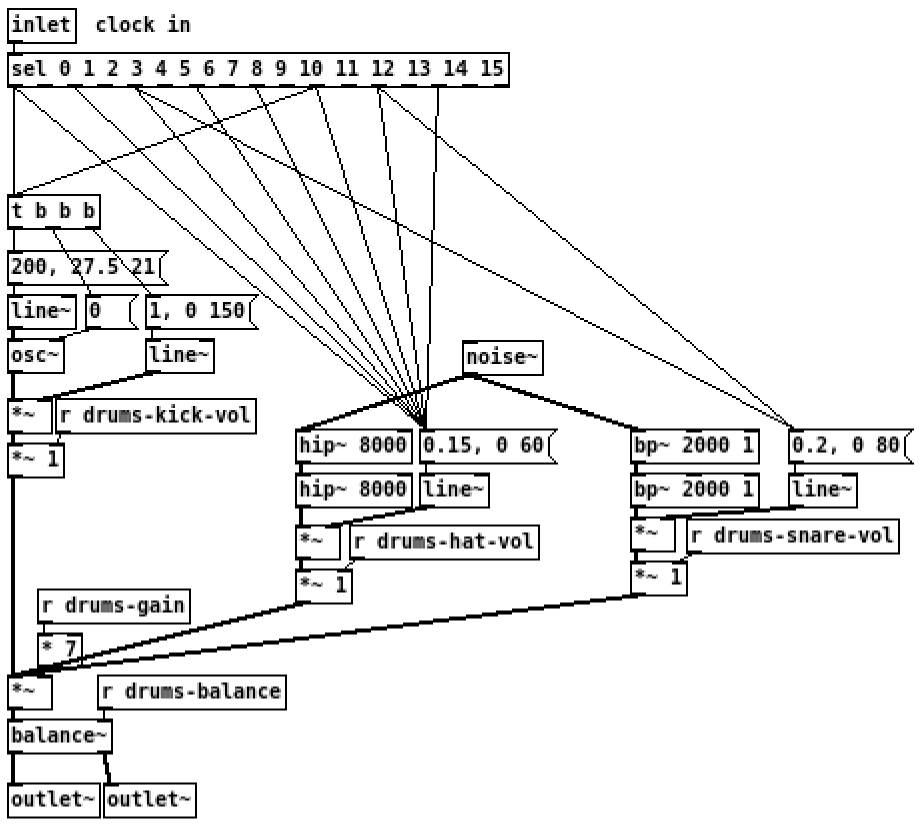

Figure A.4: `drums` subpatch.

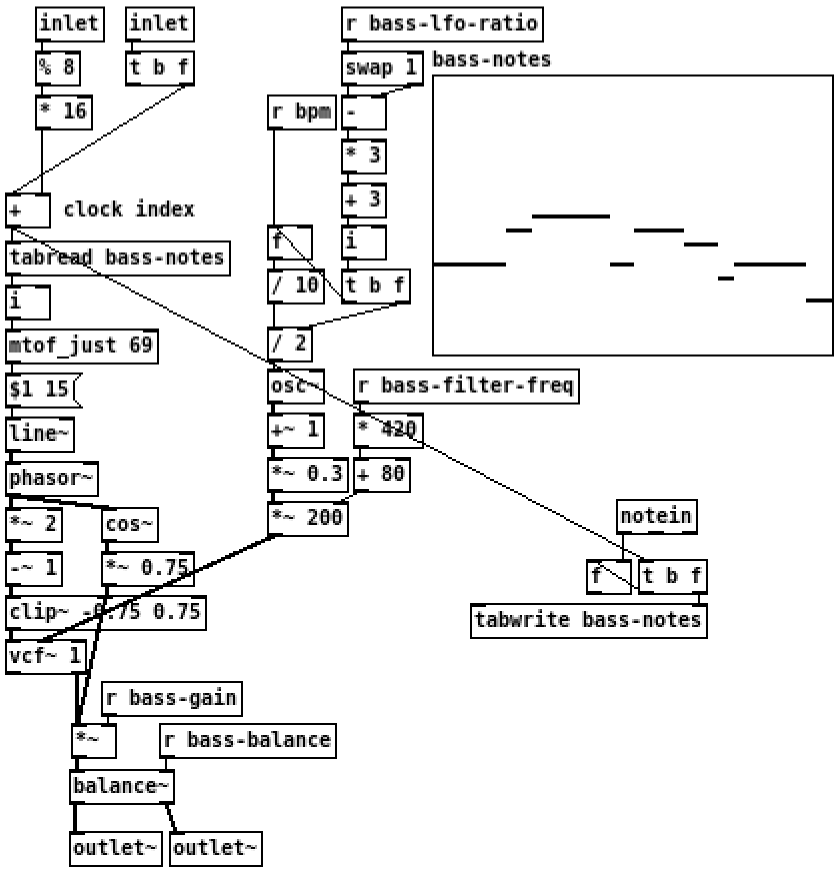

Figure A.5: `bass-line` subpatch.

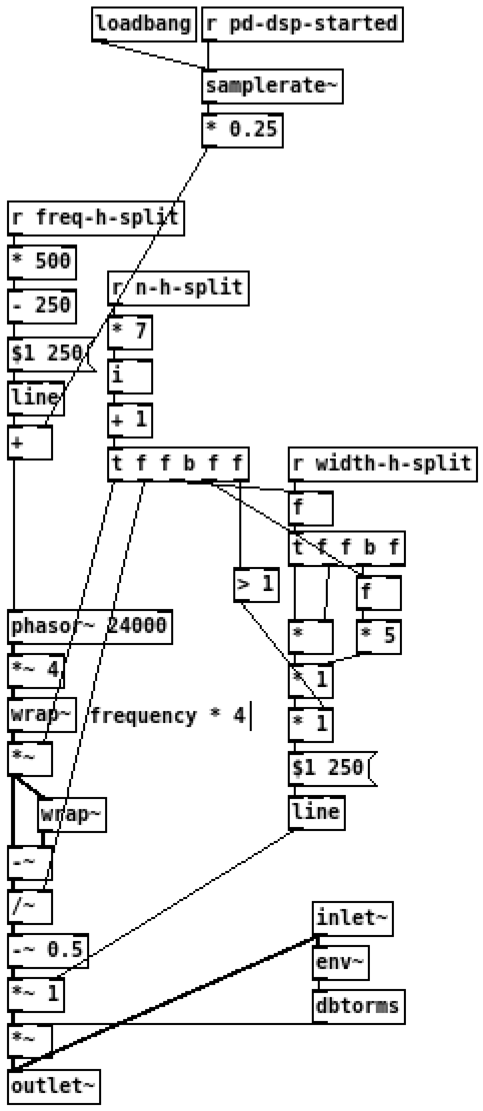

Figure A.6: subpatch creating high frequency stepped waveform.

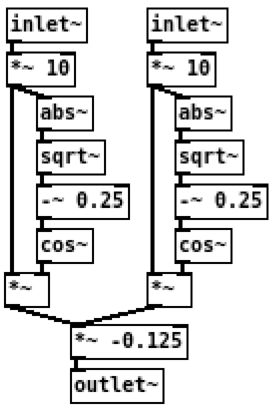

Figure A.7: Pd patch calculating Schwefel's Function No. 2.26.

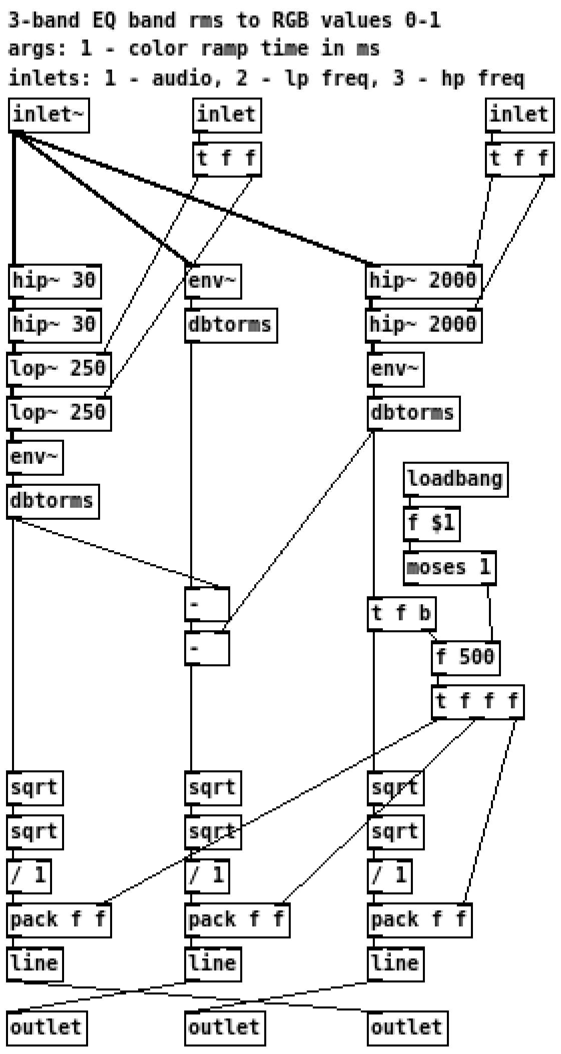

Figure A.8: Pd `speccolor~` abstraction that generates a stream of RGB values from an audio signal.

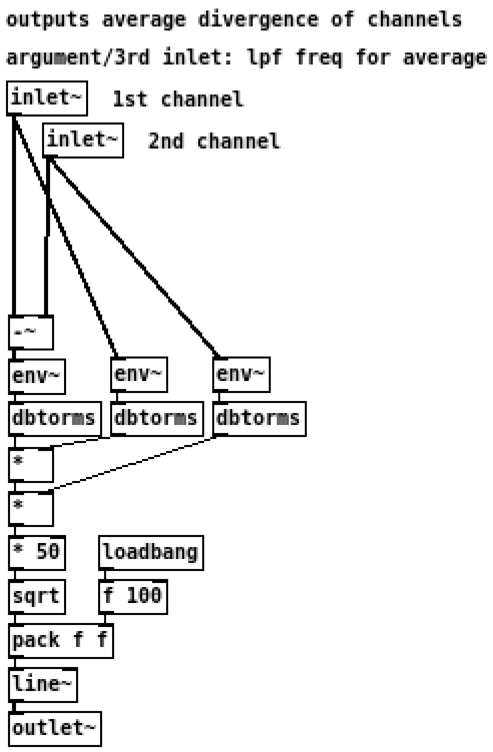

Figure A.9: Pd `avgdiff~` (stereo divergence) subpatch.

Appendix B

Accompanying Files

Path: /

mlehnfeld-thesis2017.pdf	Master thesis
mlehnfeld-demo.mp4	Video Demonstration
mlehnfeld-abstract.pdf	Abstract in English and German
mlehnfeld-project.zip	Pure Data files

Path: mlehnfeld-project.zip/

aaron~.pd	Audio synthesis module
adsr.pd	Envelope abstraction from the Pd documentation
balance~.pd	Stereo balance abstraction
eden.pd	Main patch for the audio process
fador~.pd	Oscillator with three basic wave shapes
mtof_just.pd	Converts MIDI indices to frequencies according to a 7-limit tuning
playnote~.pd	Single voice abstraction for polyphonic synthesizer
speccolor~.pd	Generates RGB values from an audio stream
terrainlookup~.pd	Implements the arithmetic terrain function
terrain~.pd	Performs wave terrain distortion on one audio channel
visuals.pd	Main patch for the visualization process

References

Literature

Arita, M, and S Iwamiya. 2004. “The effects of structural characteristic of switching patterns and pitch patterns of sound on the subjective congruency of sound and moving picture”. In *Proceedings of the Autumn Meeting of Japanese Society of Music Perception and Cognition*, 71–76. (Cit. on p. 7).

Arita, M, X Su, H Kawakami, M Ueda, and S Iwamiya. 2005. “The subjective congruency between direction of pitch movement and direction of switching pattern movement”. In *Proceedings of the Autumn Meeting of Japanese Society of Music Perception and Cognition*, 31–36. (Cit. on p. 7).

Borgonovo, Aldo, and Goffredo Haus. 1986. “Sound synthesis by means of two-variable functions: experimental criteria and results”. *Computer Music Journal* 10 (3): 57–71. (Cit. on p. 25).

Caivano, José Luis. 1994. “Color and sound: Physical and psychophysical relations”. *Color Research & Application* 19 (2): 126–133. (Cit. on p. 5).

Ciciliani, Marko. 2016. “An introspective method for the analysis of musical multimedia”. (Cit. on p. 8).

Danks, Mark. 1997. “Real-time Image and Video Processing in GEM.” In *Proceedings, International Computer Music Conference.* (Cit. on p. 20).

Erlich, Paul. 1998. “Tuning, tonality, and twenty-two-tone temperament”. *Xenharmonikon* 17:12–40. (Cit. on p. 10).

Evans, Karla K, and Anne Treisman. 2009. “Natural cross-modal mappings between visual and auditory features”. *Journal of vision* 10 (1): 6–6. (Cit. on p. 5).

Ferguson, Sam, Andrew Vande Moere, and Densil Cabrera. 2005. “Seeing sound: Real-time sound visualisation in visual feedback loops used for training musicians”. In *Information visualisation, 2005. proceedings. ninth international conference on*, 97–102. IEEE. (Cit. on p. 13).

Fourney, David W, and Deborah I Fels. 2009. “Creating access to music through visualization”. In *Science and Technology for Humanity (TIC-STH), 2009 IEEE Toronto International Conference*, 939–944. IEEE. (Cit. on p. 12).

Hinrichsen, Haye. 2016. “Revising the musical equal temperament”. *Revista Brasileira de Ensino de Física* 38 (1). (Cit. on p. 10).

Honingh, Aline Klazina. 2006. “The origin and well-formedness of tonal pitch structures”. PhD thesis, Institute for Logic, Language and Computation, Universiteit van Amsterdam. (Cit. on p. 10).

Hubbard, Timothy L. 1996. “Synesthesia-like mappings of lightness, pitch, and melodic interval”. *The American journal of psychology*: 219–238. (Cit. on p. 6).

Iwamiya, Shin-ichiro. 2013. “Perceived congruence between auditory and visual elements in multimedia”. *Tan et al., Psychology of Musical Multimedia*: 141–164. (Cit. on pp. 6–8).

Kim, K H, S Iwamiya, and H Kitano. 2008. “Subjective congruence between rotating transition of moving picture and pitch shift pattern of a sound”. In Japanese, *Journal of Music Perception and Cognition*, no. 14: 29–36. (Cit. on p. 7).

Lipscomb, Scott D, and Eugene M Kim. 2004. “Perceived match between visual parameters and auditory correlates: an experimental multimedia investigation”. In *Proceedings of the 8th International Conference on Music Perception and Cognition*, 72–75. (Cit. on p. 7).

Maeda, Fumiko, Ryota Kanai, and Shinsuke Shimojo. 2004. “Changing pitch induced visual motion illusion”. *Current biology* 14 (23): 990–991. (Cit. on p. 7).

Marks, Lawrence E. 1974. “On associations of light and sound: The mediation of brightness, pitch, and loudness”. *The American journal of psychology*: 173–188. (Cit. on p. 6).

— . 1987. “On cross-modal similarity: Auditory–visual interactions in speeded discrimination.” *Journal of Experimental Psychology: Human Perception and Performance* 13 (3): 384. (Cit. on pp. 5, 6).

Palmer, Stephen E, Karen B Schloss, Zoe Xu, and Lilia R Prado-León. 2013. “Music–color associations are mediated by emotion”. *Proceedings of the National Academy of Sciences* 110 (22): 8836–8841. (Cit. on pp. 5, 6).

Parise, Cesare V, and Charles Spence. 2012. “Audiovisual crossmodal correspondences and sound symbolism: a study using the implicit association test”. *Experimental Brain Research* 220 (3-4): 319–333. (Cit. on p. 5).

Patel, Neel V. 2015. “Symphony for Oscilloscope”. *IEEE Spectrum* 52 (3): 26–26. http://spectrum.ieee.org/consumer-electronics/audiovideo/jerobeam-fendersons-trippy-oscilloscope-music. (Cit. on p. 15).

Peacock, Kenneth. 1988. "Instruments to perform color-music: Two centuries of technological experimentation". *Leonardo* 21 (4): 397–406. (Cit. on p. 12).

Puckette, Miller. 1996a. "Pure Data: another integrated computer music environment". *Proceedings of the second intercollege computer music concerts*: 37–41. (Cit. on p. 18).

Puckette, Miller S. 1996b. "Pure Data." In *Proceedings, International Computer Music Conference*, 224–227. San Francisco: International Computer Music Association. (Cit. on pp. 18, 19).

Recanzone, Gregg H. 2009. "Interactions of auditory and visual stimuli in space and time". *Hearing research* 258 (1): 89–99. (Cit. on pp. 4, 5).

Rusconi, Elena, Bonnie Kwan, Bruno L Giordano, Carlo Umilta, and Brian Butterworth. 2006. "Spatial representation of pitch height: the SMARC effect". *Cognition* 99 (2): 113–129. (Cit. on p. 7).

Schwefel, Hans-Paul. 1981. *Numerical Optimization of Computer Models.* (Cit. on p. 27).

Shannon, Claude Elwood. 1949. "Communication in the presence of noise". *Proceedings of the IRE* 37 (1): 10–21. (Cit. on p. 14).

Simpson, Ray H, Marian Quinn, and David P Ausubel. 1956. "Synesthesia in children: Association of colors with pure tone frequencies". *The Journal of genetic psychology* 89 (1): 95–103. (Cit. on p. 6).

Stine, Eli. "Sound Vision 2.0: Bark Scale Visualizer". (Cit. on p. 12).

Ward, Jamie, Brett Huckstep, and Elias Tsakanikos. 2006. "Sound-colour synaesthesia: To what extent does it use cross-modal mechanisms common to us all?" *Cortex* 42 (2): 264–280. (Cit. on p. 6).

Online sources

"APEXvj = Good vibes". 2017. https://www.apexvj.com/v3/. (Cit. on p. 13).

Bown, Ollie, Benito Crawford, Ben Porter, and Aengus Martin. 2017. "The Beads Project - Realtime Audio for Java and Processing". http://www.beadsproject.net/. (Cit. on p. 18).

"Butterchurn visualizer". 2017. https://butterchurnviz.com/. (Cit. on p. 13).

Cooke, Jethro. 2016. "Asterism 2.0". https://player.vimeo.com/video/153341279. (Cit. on pp. 16, 24).

Di Fede, Damien, and Anderson Mills. 2017. "Minim". http://code.compartmental.net/minim/. (Cit. on p. 18).

Encyclopædia Britannica, Inc. 2017. “Lissajous figure | Britannica.com”. https://www.britannica.com/topic/Lissajous-figure. (Cit. on p. 14).

Fenderson, Jerobeam. 2014. “How To Draw Mushrooms On An Oscilloscope With Sound”. https://www.youtube.com/watch?v=rtR63-ecUNo. (Cit. on p. 24).

— . 2015. “Jerobeam Fenderson”. http://www.jerobeamfenderson.net/tagged/puredata. (Cit. on p. 15).

— . 2017a. “Oscilloscope Music by Jerobeam Fenderson — Kickstarter”. https://www.kickstarter.com/projects/1230242223/oscilloscope-music/. (Cit. on p. 15).

— . 2017b. “Oscilloscope Music by Jerobeam Fenderson — Kickstarter”. https://www.kickstarter.com/projects/1230242223/oscilloscope-music/posts/1791562. (Cit. on p. 15).

Fry, Ben, and Casey Reas. 2017. “Processing.org”. https://processing.org/. (Cit. on p. 18).

“Gibber”. 2017. http://gibber.cc/. (Cit. on p. 13).

Higinbotham, William. 2017. “Tennis for two”. http://www.sunysb.edu/libspecial/videogames/tennis.html. (Cit. on p. 14).

“Music Visualizer, VJ Software & Beyond: Magic Music Visuals”. 2017. https://magicmusicvisuals.com/. (Cit. on p. 13).

“p5.js | home”. 2017. https://p5js.org/. (Cit. on p. 13).

“Plane9 a scene based music visualizer and sound responsive screensaver”. 2017. https://www.plane9.com/. (Cit. on p. 13).

Puckette, Miller. 2006. “Making and using all-pass filters”. http://msp.ucsd.edu/techniques/v0.11/book-html/node161.html. (Cit. on p. 28).

“Synesthesia - Live Music Visualizer - VJ Software”. 2017. http://synesthesia.live/. (Cit. on p. 13).

“three.js - Javascript 3D library”. 2017. https://threejs.org/. (Cit. on p. 13).

Tone.js. 2017. “Mann, Yotam”. https://tonejs.github.io/. (Cit. on p. 18).

Väänänen, Pekka. 2014. “Quake on an oscilloscope: A technical report”. http://www.lofibucket.com/articles/oscilloscope_quake.html. (Cit. on p. 14).

“VSXu - audio visualizer, music visualizer, visual programming language (VPL), realtime graphics design platform”. 2017. http://www.vsxu.com/. (Cit. on p. 13).

“vvvv - a multipurpose toolkit | vvvv”. 2017. https://vvvv.org/. (Cit. on p. 13).