PORTFOLIO OF COMPOSITIONS

by

HELENE HEDSUND

A thesis submitted to

The University of Birmingham

for the degree of

DOCTOR OF PHILOSOPHY

Department of Music

College of Arts and Law

The University of Birmingham

September 2019

ABSTRACT

This is a portfolio of six electroacoustic multichannel pieces created between 2013 and 2019. They illustrate chronologically my desire to completely stop using Digital Audio Workstations (DAWs) and only work with my own software tools. I call these software tools instruments/composition tools because they are both. I have a background as a software developer so such a wish comes naturally to me, having enjoyed building large code structures before. I find it an intuitive way to develop a composition. When I started this PhD project I was not able to produce a musical piece without some use of a DAW but now I can, so this text also illustrates my learning curve. Working within DAWs imposes certain limitations and assumptions upon the user. Working with instruments allows greater flexibility both in the compositional process, and in creating alternative versions, for example for different speaker layouts. That said, in keeping with the nature of the programme at Birmingham, the main focus of this portfolio is the compositions themselves. It is not a detailed investigation of different workflows. However, I hope that it might serve as an example and inspiration for other composers to reflect on their own workflows and perhaps inspire them to build their own toolsets that go beyond DAWs and other standard tools. The compositions themselves serve as a concrete example of such workflows employed in practice. In a broader sense, these instruments/compositions are one contribution to a renewed surge of interest in the use of code and custom software in the creation of musical compositions, which has arisen in part in response to the limitations presented by commercial software.

An additional area of exploration of the ‘unmasking’ of previously masked sounds in recordings as a compositional strategy, revealing hidden aspects of a sound’s spectrum. This is employed in two of the pieces in the portfolio.

Finally, in musical terms they were designed with certain symbolic aspects such as the choice of durations, and the dates on which the recordings were made. These symbolic aspects and choices relate to my mother, to whom the portfolio is dedicated.

To my mother who made it possible

Inger Rikand (1944-2012)

ACKNOWLEDGEMENTS

I would like to thank my supervisor Dr. Scott Wilson and also Annie Mahtani at the Music Department at University of Birmingham. I would also like to thank the institutions that have given me financial support:

- Helge Ax:son Johnson
- Konstnärsnämnden
- University of Birmingham
- STIM

Also, a very big thank you to my partner Magnus Johansson, my aunt Gunnel Minett and Elektronmusikstudion (EMS) in Stockholm, Sweden.

LIST OF ABBREVIATIONS

DAW Digital Audio Workstation.

EMS Elektronmusikstudion in Stockholm. http://www.elektronmusikstudion.se.

GUI Graphical User Interface.

MOD. “MODULUS In computing, the **modulo** operation finds the remainder after division of one number by another (called the *modulus* of the operation).” Source: https://en.wikipedia.org/wiki/Modulo_operation. August 20 2019.

RQ “The reciprocal of Q. Q is conventionally defined as cutoffFreq / bandwidth, meaning rq = (bandwidth / cutoffFreq).” Source: http://doc.sccode.org/Classes/BPF.html, August 12, 2019.

SDIF "’Sound Description Interchange Format’ is a standard for the well-defined and extensible interchange of a variety of sound descriptions. SDIF consists of a fixed framework plus a large and extensible collection of spectral description types, including time-domain (analogous to regular audio file formats), Sinusoidal Models, other spectral models, and higher-level models. SDIF was jointly developed by IRCAM and CNMAT.” Source: https://en.wikipedia.org/wiki/SDIF, August 20 2019.

12-ET “In classical music and Western music in general, the most common tuning system since the 18th century has been twelve-tone equal temperament (also known as 12 equal temperament, 12-TET or 12-ET), which divides the octave into 12 parts, all of which are equal on a logarithmic scale, with a ratio equal to the 12th root of 2 (12√2 ≈ 1.05946).”Source: https://en.wikipedia.org/wiki/Equal_temperament, August 12, 2019.

LIST OF ILLUSTRATIONS

LIST OF TABLES

USB STICK CONTENT

The submitted USB stick contains these directories and contents:

- HH/
    - appendices/
        - images/
            - guis/ – big images of the instrument GUIs.
            - sonograms/ – big uncropped sonogram images.
        - otherVersions/ - other versions of three of the submitted pieces.
        - rp/ – the sound examples discussed in the appended Research Proposal.
    - HH.pdf – this text.
    - instruments/ – the six instruments and their source sounds.
    - pieces/ – all pieces in this portfolio in interleaved multichannel format.

TABLE OF CONTENTS

1. INTRODUCTION

My mother gave birth to her first and only child on December 9, 1963. She married the love of her life on December 9, 1978. She died on December 9, 2012. All work described in this portfolio has been created with her in mind. Therefore I have used the numbers 9 and 12 in all of the pieces. For example, all pieces have a duration of 12:09 or 9:12, this portfolio contains nine pieces, in the piece *NINE STATIONS* there are sounds from nine railway stations and the sounds are filtered through nine frequencies. I have also used recordings done on December 9, 2013, 2014, 2015, 2016 and 2017.

I started my musical activities playing in post punk bands. After that I studied all music subjects there were at the universities near me. Musicology, Counterpoint, Music Psychology, Instrumental composition etc. In 1994 I found EMS, Elektronmusikstudion in Stockholm, Sweden. In 1996 I got a travel scholarship from Uppsala Universitet. I chose to go to the UK and study composition with Justin Connolly[1]. In 1997 I went to City University to study a combination of EAM and instrumental composition. This did not turn out well so I went home and gave up music for seven years which I spent studying IT and programming and later working professionally as a software developer. After a concert with my childhood favourite band Sparks[2] I decided that it was time to return to music, so in 2005 I went back to EMS. During the period 2008-2010 I studied both EMS two year course and took a Masters Degree in Sound Art at Dramatiska Institutet in Stockholm.

I am both a composer and a programmer. I don’t like to make my music in DAWs (Digital Audio Workstation), the piece of software commonly used for composition and editing in digital music studios. When I use them I always end up with many gigabytes of sounds, and it very easily

[1] Justin Connolly is a British composer and former teacher at Royal College of Music and Royal Academy of Music. See: https://en.wikipedia.org/wiki/Justin_Connolly. August 28 2019.

[2] “Sparks are an American pop and rock band formed in Los Angeles in 1967 by brothers Ron (keyboards) and Russell Mael (vocals).” Source: https://en.wikipedia.org/wiki/Sparks_(band). August 28 2019.
Sparks albums *Kimono My House* and *Propaganda* from 1974 were constantly played in my room for a long time.

becomes messy. If I work with code I can keep my projects small and tidy; a directory with sound files and a few text files with code – i.e. the instructions for how to turn the sound files into music. One of my goals during the years I have been working on this portfolio has been to get away from DAWs.

My musical interests are not primarily in electroacoustic music. I work in electroacoustic studios but mostly as a means to an end. When I found EMS I was thrilled but not because I liked the electroacoustic music I was introduced to. My enthusiasm arose when I realized that from now on I could “do everything myself”. I would not need musicians any more. Also, coming from a rock background I was already used to electronic instruments and sound effects like guitar boxes and tape echoes. I also had experience from analogue music studios of the kind used while recording records and demos. At EMS I found a new kind of studio that I could operate myself. Wonderful! So, coming from playing in bands, then continuing with studies in instrumental music and then finding EMS, I suddenly became an electroacoustic composer – simply because I worked in an electroacoustic studio, and because my output hardly could be called popular or commercial. I call myself an electroacoustic composer because I work with electroacoustic tools and material, but within I consider me to make music. My music without a label.

As is obvious from the rest of this text, I use filters a lot. Once I got an assignment to analyse John Chowning’s piece *Stria*. There is a lot of material about this piece online, for example *The Reconstruction of Stria (*Computer Music Journal, 31:3, 2007). I decided to make my own version of *Stria*, an endeavour that later led to a piece of mine called *Skrika*. Skrika is Swedish and means ‘to scream’. Instead of FM synthesis I used recordings of people screaming and filters according to Chowning’s golden mean *Stria* scale. The result sounded very much like *Stria*. I got very fond of filters and scales and I have used both a lot since then.

Many years ago, I dreamed of music. I was walking in a forest and heard something that sounded like a string orchestra. It was very beautiful. Suddenly the music changed, it started to skew. The strings’ overtones became inharmonic and it resulted in something like auditory vertigo. I have of course tried to recreate this skewing effect, but have not yet succeeded fully. My piece *INANNAN* is the closest I have come. This is the reason for my interest in spectrums and partial analyses and for my experiments with simultaneously playing several versions of the same sound source, e.g. an actual recording and its partial analysis, and to go from the actual to something manipulated. I want to hear the change, just like in my dream.

1.1 RESEARCH PROPOSAL

In my original research proposal from 2013 I wrote that I wanted to:

1. Unmask previously masked features in recordings.
2. Continue work on my first generic instrument.

I had worked a lot with the first proposal before and found it interesting. For example, I recorded musicians playing their instruments. I removed the loud instrument sounds from the recording and kept only the soft noises the musician did while playing. I found it interesting to be able to hear sounds that were not possible to hear otherwise. Examples of this interest in this portfolio is the laundry rack sounds in *BUS NO. 1,* and the characteristic sounds in *NINE STATIONS.* The unmasking is described (with sound examples) in my research proposal which I have included in the APPENDIX.

I have also worked with proposal number two. This work led to my second generic instrument, but after *BUS NO. 1* I began to write piece dedicated instruments instead. I will however, return to my generic instrument in the future. Part of my goal during these PhD years have been to learn how to program my own tools and become independent of DAWs and plugins.

1.2 CONTEXT

The composer Tamas Ungvary did many things that I do as early as 1980. Tamas Ungvary worked at EMS for many years, especially with the computer controlled analogue/digital sound system designed in the early 1970's and shut down around 1990. I have read many of his texts, some published, some not. For instance, he developed methods for working with sonograms. He considers a sonogram to be a "*Visual documentation of structural information: a tool for composition and analysis in electro-acoustic music*." (Ungvary and Waters 1988, p. 1)[3] . He also uses the computer keyboard as an instrument as can be seen in a video online[4]. In the video he demonstrates his FORTRAN program for EMS's computer: CHOR. Tamas Ungvary has made many software instruments, the latest one uses a Sentograf (a pressure sensitive ball) to enable agogics.

The following text will describe how I have turned previously fixed pieces into live instruments. I'm not the only one who has done this. Jean-Michel Couturier describes in his paper: *ETHERACTION: PLAYING A MUSICAL PIECE USING GRAPHICAL INTERFACES (2004)* how he "*designed two interactive systems dedicated to interpret in live Etheraction, a multichannel piece*" which he *"initially composed for tape*." (Couturier 2004, p1)[5]

Michael Edwards has written the paper S*lippery chicken a specialised algorithmic composition program*. He writes that: *"The main goal of the project is to facilitate a melding of electronic and instrumental sound worlds, not just at the sonic but also at the structural level"* (Edwards, M. 2012. p. 1)[6]. I think that this resembles my experiments with trying to extract scales from field recordings as I did in my piece *NINE STATIONS*.

[3] Tamas Ungvary and Simon Waters. (1988). REPORTS K.A.C.O.R. Visual documentation of structural information : a tool for composition and analysis in electro-acoustic music, 1/88. Page 1.

[4] https://www.youtube.com/watch?v=Jqmd6x5-7n8 (August 29, 2019)

[5] Jean-Michel Couturier. (2004). ETHERACTION: PLAYING A MUSICAL PIECE USING GRAPHICAL INTERFACES.

[6] Michael Edwards. (2012). slippery chicken a specialised algorithmic composition program.

In the BIBLIOGRAPHY I have listed all the texts I have read while writing this text, even though I have not cited them directly. I found many of them while looking for papers on how to make your own software instruments.

There are common problems for many who work with building their own software instruments. Some examples are:

When using the computer keyboard as a controller, there are a limited set of keys to use, so in some cases you have to choose which parameters to control from the keyboard, and perhaps invent.other means to control other parameters. Also, one has to think about what values are practical to use when changing a parameter. In one case it might be practical to increment/decrement with +- 10, in another it might be practical to use various lists, for example [1, 2, 3] or [5, 10, 15], in yet other cases it might be better to use descriptive words like low, mid, high.

There can also be a problem with GUI space. It is often not possible to give a full description of what for example a button does, and an abbreviation might not be easy to understand (after not using the instrument for a while) even if I made it up myself. Here are two examples of mappings between available keys to functionality. The first one from Tamas Ungvary’s user manual for his program ILI (Interactive List Interpreter), the second one is my own.

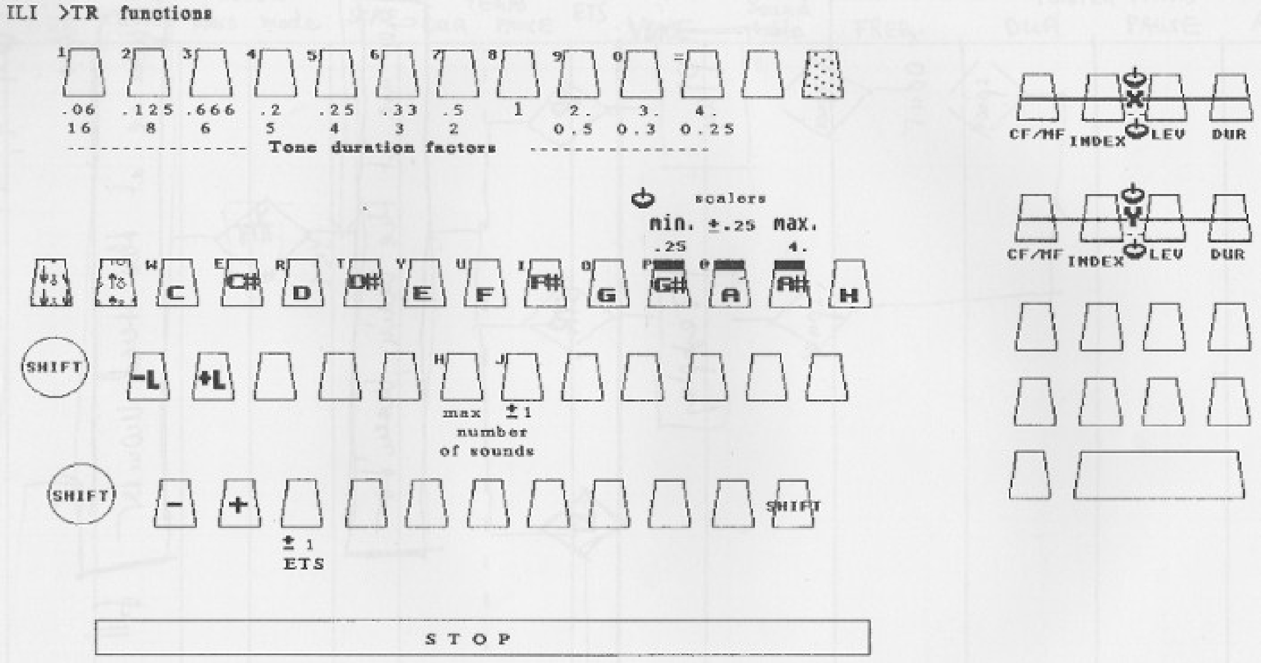
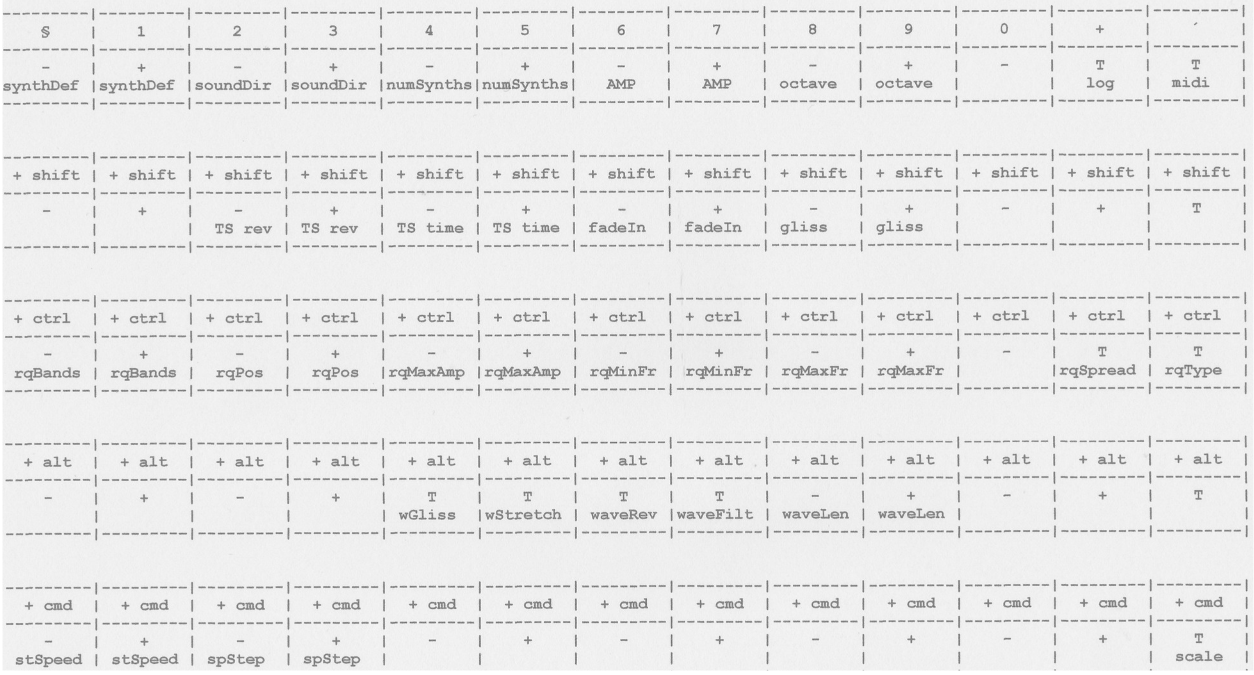


Figure 1: Keyboard mapping from Tamas Ungvary's ILI (Interactive Llist Intrepreter) user manual.



§	1	2	3	4	5	6	7	8	9	0	+	´
- synthDef	+ synthDef	- soundDir	+ soundDir	- numSynths	+ numSynths	- AMP	+ AMP	- octave	+ octave	-	T log	T midi

+ shift	+ shift	+ shift	+ shift	+ shift	+ shift	+ shift	+ shift	+ shift	+ shift	+ shift	+ shift	+ shift
-	+	- TS rev	+ TS rev	- TS time	+ TS time	- fadeIn	+ fadeIn	- gliss	+ gliss	-	+	T

+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl	+ ctrl
- rqBands	+ rqBands	- rqPos	+ rqPos	- rqMaxAmp	+ rqMaxAmp	- rqMinFr	+ rqMinFr	- rqMaxFr	+ rqMaxFr	-	T rqSpread	T rqType

+ alt	+ alt	+ alt	+ alt	+ alt	+ alt	+ alt	+ alt	+ alt	+ alt	+ alt	+ alt	+ alt
-	+	-	+	T wGliss	T wStretch	T waveRev	T waveFilt	- waveLen	+ waveLen	-	+	T

+ cmd	+ cmd	+ cmd	+ cmd	+ cmd	+ cmd	+ cmd	+ cmd	+ cmd	+ cmd	+ cmd	+ cmd	+ cmd
- stSpeed	+ stSpeed	- spStep	+ spStep	-	+	-	+	-	+	-	+	T scale

Figure 2: My own mapping between keyboard and functionality used in most of my instruments.

Another aspect that easily comes to mind when using the computer keyboard as interface is tuning, since you can tune the non piano like keyboard in any way you choose. I have spoken to Tamas Ungvary about this and know that we both share this interest in changing the keyboard's tuning.

1.3 INSPIRATION AND INFLUENCES

I am an introverted person. I like to work on my own with my own projects. I have studied almost all of my life so I have of course been influenced by very much. In hindsight the strongest influences have been:

- Justin Connolly showed me how I could use pen and paper to draw maps or graphs, time lists, shapes and what ever that could help me with my compositions if I got stuck. This technique is something that I still use today.
- I have heard talks by Michael Edwards, Tamas Ungvary and Fernando Lopez-Lezcano and they have all inspired me to make my own composition tools. I was inspired by the fact that they made their own software tools, not by any specific features in their instruments.
- During the SuperCollider Symposium in Berlin 2010 I saw a sound installation by Jost Muxfeldt[7]. It was a video of a virtual mobile with lots of parts moving in circles. A sound was attached to each of the moving nodes. The sounds were spread out in a circle round the listener and the form of the music was made up by these moving nodes making the sounds fade in and out and move around the room. I think that what I found most interesting was that I could sort of see the musical form. I found it fascinating and looked at it for a long time, but until this day I have not found a way to evolve this inspiration into something of my own. I have thought about using it for spatialisation. Perhaps I could use some type of drawing pad to draw shapes on. Start a sound that spatially moves in the trajectory I have drawn. See it moving and then start a new sound in another drawn trajectory, and so on. That

[7] Parts of the installation can be seen here: https://www.youtube.com/watch?v=0cVIySdDXs4 (October 4, 2020). A description of the project can be found here: http://kinematics.glyph.de/en/project/audiokinematics

would resemble Jost Muxfeldt’s installation but without his interesting mobile part. But I think that this would only be an interesting experiment and not a very usable composition tool.

- In 1997 I attended a diffusion course at EMS. Dennis Smalley was our teacher. My enthusiasm arose on the last day of the course when we had the opportunity to diffuse our own music at Fylkingen concert space. I don't remember how many speakers we used but I remember that we could diffuse our stereo pieces with a joystick. Suddenly my music was everywhere!! After that day multichannel was what I wanted to do but I had to wait 11 years before I got the opportunity to do so.
- Benjamin Thigpen held a course on AudioSculpt at EMS in 2008. This was a big revelation to me and I have used AudioSculpt in all my pieces ever since. This was a long time ago so I don’t remember exactly what it was that I found so interesting, but it had to do with the sonograms which enabled me to both see and manipulate a sound. If there was an annoying high beep in a recording I could remove it. If there was too much low bass I could remove it. I could filter out chosen frequencies. I could remove loud frequencies thus enabling me to hear a sound’s soft components. After that course (probably) all my sound files have been checked in AudioSculpt to see if there are any disturbing noises in them.

1.4 CONTRIBUTION TO THE FIELD?

In this portfolio of compositions the compositions themselves are the important part. The instruments discussed in this supplementary text were made to make the pieces, not the other way round. Still, my instruments might be an unusual approach to composing and performing electroacoustic music and might serve as an inspiration to other composers. Using my instruments is a different approach to composing compared to making fixed media pieces in a DAW. As I can’t see a graphical representation of the music when composing with my instruments, it encourages more of listening and “playing” my way forward instead of placing sound files on a DAW timeline.

Another way of using the instruments could be during concerts. It has still not happened, but I think that it would be very interesting to hear someone else play my pieces on my instruments. Would this other person play it in a way that I hadn't thought of myself? Would this improve the piece? Also, instead of just sending my sound files to a concert, I could send an instrument. That would require that someone was prepared to practice and then play a piece and that may never happen, but I find the idea interesting. Having the piece interpreted by a performer would be an alternative way to play electroacoustic pieces in concerts in the future.

A practical example for how the instruments can be used by other people than me: For a few years now, I have had a collaboration with the composer and cantor Staffan Björklund. He has many, many years of experience in composing instrumental music. Some years ago he decided that he wanted to make an "electroacoustic" piece based on an orchestral piece he wrote in the 1970s. He knew approximately what he wanted to achieve but felt that his eyesight was too poor and that he lacked some technical skills, so he contacted me to help him. While working together Staffan has explained to me what types of sounds and effects he wants. My intention is to build him an instrument which will be able to produce the sound world he desires, but as it will be an instrument, I will not compose the piece for him. I think that this is a practical solution to Staffan’s problem. I imagine that other people in a similar situation would also benefit by having instruments built for them.

In a broad sense my instruments can be seen as a further development in a resurgence of interest in developing custom software tools for composition, perhaps in response to the limitations imposed by commercial software (Blackwell & Collins 2005). It is thus related to both recent developments such as live coding (Zmölnig & Eckel, 2007), and early software instruments such as Truax’s POD system (Truax, 1988), though in a form which exploits many of the modern possibilities of an interpreted real-time language.

1.5 INDEPENDENCE

I have a strong desire to be independent. This is the reason for my happiness after finding the electroacoustic studio: I realised that from now on I could be independent of musicians and other persons like studio technicians. I would be able to do everything myself.

This is also the reason for my reluctance to be dependent on complex pieces of software, for example a DAW, which has not even been developed primarily with the electroacoustic composer’s needs in mind. New versions of the software come out and sooner or later I will have to buy the new version as the old one will stop working. Things in the software will have changed, I will have to learn anew, which might not be what I want. When developing my own tools I have better control over my pieces.

Avoiding dependency on commercial decisions made by closed source software vendors is also a reason why I since 1998 have been using open source software, mostly running Linux. My first instrument was developed on Linux, but during the years that I've been pursuing this PhD I have frequently worked in studios where the sound cards have not been compatible with Linux, so I have mostly used MacOS X. I will however return to Linux as soon as I can.

This also explains why I avoid using SuperCollider quarks or extensions, and stay with the core parts of the SuperCollider language. I want to have control over my own tools. If other people have developed and maintain code parts which are vital to my code, the risk that things stop working at the next software upgrade increases, and I lose my independence.

2. MY INSTRUMENTS

2.1 WHY INSTRUMENTS?

I always change my pieces (and most other things in my surroundings). While listening to my own piece in a concert situation, I almost always think of ways to improve the piece. "What if I did this.. perhaps a little louder here... something is missing here". So, on the next performance of the piece I will have made a new version with the changes I thought of during the last concert. I think that I somehow borrow the ears of my fellow audience. I automatically try to put myself in their shoes, listening to the piece for the first time, and I hear it differently, from a more detached angle. One purpose of my instruments/composition tools is to let me easily make a new interpretation of the piece for the next performance. Unlike many composers, I generally do not consider my pieces to ever be completely ‘finished’. They are always subject to possible improvements.

The instruments are also a way to escape the agony of having to make so many decisions while composing. In 1997, when I temporarily gave up music and started studying programming, what I liked most was that in programming there was actually (comparatively) a right and a wrong. If my employer wanted me to write functionality that performed a certain task, it was easy to check if my code did what it was supposed to do or not. When composing I always suffered from decision anxiety. Should I do it like this? Or like that? Should this come before that? Or the other way around? My instruments/composition tools are a way for me to not have to decide. I can do it like this today and do it differently tomorrow.

In contrast, the workflow when using a DAW is less flexible, especially considering a complex multi-channel setup. When a final version once has been rendered in a DAW, it often seems to require too much effort to go back and create a new version later.

2.2 INSTRUMENT VERSUS DAW WORK METHODS

My instruments are intended for being used as live performance instruments but also as composition tools. All piece dedicated instruments have an auto version. I made these auto versions by playing on my instruments. I tried different ideas and stored the result (in the auto version file) when I was happy with what I heard. The difference with this style of working compared to using a DAW is that it is much easier to try different combinations. It thus offers substantially increased flexibility.

Before continuing, I must admit that I began using the DAW ProTools in the 1990s and for the last ten years I have only used it as a “tape recorder”. This means that my way of working with a DAW might be a bit old fashioned. There might be new opportunities now than I’m not aware of.

In a DAW you place your sound files on a time line. They stay there until you move them. So, if you work on a multichannel piece you might end up with 80 tracks or more. Because of screen size you can't even see all the tracks at once. So lets say that you have three different types of sounds sounding at once. (A minimum of 24 tracks.) If you want to change the order of these sounds, you'll have to move all 24 sound files on the time line. If there is not free space in the tracks you might have to make temporary extra tracks. (And you may find that you have to buy a more expensive license to be able to create all those tracks! The software industry is sometimes surprisingly creative in creating arbitrary limits.)

In an instrument you just play the sounds in another order[8]. Also, if you find that you miss some type of sound, you can just write a new function/play method and try it until you are satisfied. When using plug-ins in a DAW you often (at least I did when I used plug-ins) record the sound effect which gives you even more sound files. After a while - when using a DAW - I always ended up with many gigabytes of sound and so many tracks that I couldn't see them all at the same time. Also, as is sometimes the case when working in studios where you may not be able to work from your own

[8] In my second generic instrument I use SuperCollider’s History function so I can "record" what I'm playing, save the information, perhaps edit it and later play it back, much like in a sequencer. My auto versions are first played on the instrument, but later coded as SuperCollider Tasks.

hard drive or you may have to use several versions of the DAW installed on different computers, it's the question of moving, synchronizing and backing up these very big projects.

Another advantage with my instruments is that it is much easier to adapt my pieces for different numbers of speakers. (I often move between different studios with different numbers of speakers when I compose.) As I will describe below under 3.5SPATIALISATION, I don’t use sophisticated spatialisation techniques. In most pieces I simply play many similar sounds (almost simultaneously) randomly routed to one of my available speakers. With this method I can easily change a piece for e.g. four, eight or twelve speakers just by changing a number in my code. This is not as easily done when working with a fixed media piece composed in a DAW as it often requires you to visit a studio and remix the piece.

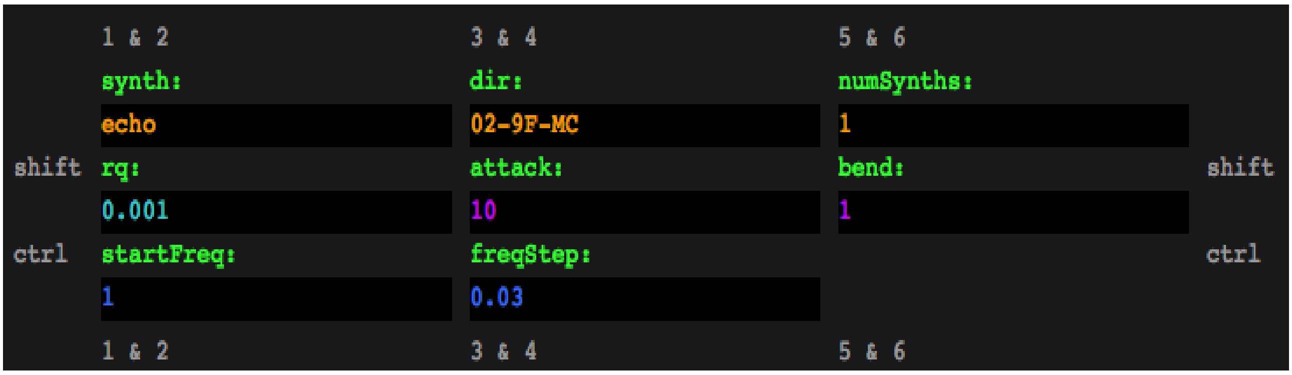
For me, composing in a DAW makes me listen and look more than think or reason. When starting on a new piece I will have a plan and I will spend a lot of time recording and editing my chosen sounds, but when I import the sounds into the DAW I start to listen my way forward. This is illustrated below by comparing the two versions of my piece *NINE STATIONS*. In the first DAW version I quickly abandoned my initial plan (partly because of deadline stress) and started to compose by listening. I don’t think that there is anything wrong with this method, I’m only reflecting on the different methods: “Composing by listening” or “Composing by reasoning, coding AND listening”. While coding I often use for example a set of frequencies in different parts of the piece and in different ways. I often think that it will sound interesting to use the same list of parameters in different sections of a piece. (Though often it doesn’t.) It is easier to play with numbers and other parameters when coding. DAWs do not necessarily make it easy to move and repurpose all forms of data. Code can be much more effective for this. It’s easier to reason your way forward as a complement – NOT as an alternative - to listening.

Another thing that differs is what you see. In a DAW you see your tracks and, at least for me, this affects my composing. When I'm coding my music I don't see a graphical representation of it and this feels uncomfortable at times. When coding I lose the comfort of seeing but enhance the possibility to think or reason. I can't really explain this difference, but to me this difference is quite clear. I reason differently when coding compared to when I use a DAW.

To summarize, creating multi-channel music using a DAW easily becomes unmanageable. In contrast, some small code files and a few directories with sound files makes the work process much easier and more flexible (and smaller in terms of gigabytes), and it offers a more structured approach where I can realize my ideas more easily.

2.3 BRIEF DESCRIPTION OF MY INSTRUMENTS

I have many instruments and they all share common features. I use the computer keyboard as controller. I use the numerical keys and four modifier keys, which gives me 65 different controls. To make playing the instrument simpler, the first and last row in the GUI list the numerical keys to use to change a value. The left- and rightmost columns list which modifier key to use. Buttons with black background colour can be changed, buttons with grey background are unchangeable, for information only. I use colours to group functionalities that belong together. When there are two numerical values for one control item they are: left - diminish, right - increase. Active buttons are highlighted.



	1 & 2	3 & 4	5 & 6	
	synth:	dir:	numSynths:	
	echo	02-9F-MC	1	
shift	rq:	attack:	bend:	shift
	0.001	10	1	
ctrl	startFreq:	freqStep:		ctrl
	1	0.03		
	1 & 2	3 & 4	5 & 6	

Figure 3: Example instrument

The instruments have three types of buttons:

- Step button, a button that when pressed increases its value one step and when it reaches its maximum it goes back to its beginning again. E.g. 0, 1, 2, 3, 4, 0, 1, 2 etc.
- Minus and plus (-+) button pairs, two adjacent buttons where the left button decreases its value and the right button increases it.
- Toggle button, a button that toggles something on or off.

The instruments have three types of text fields:

- Value field, a field that shows an item’s current value. These fields have a black background colour.
- Information field, a field that is for information only. Its value can’t be changed by the user. These fields have a grey background colour.
- Timer field, a field that is for information only and shows for how long something has been running. These fields have a grey background colour.

If you want to try any of the instruments in this portfolio you should know this:

They have been confirmed to work on MacOS X with SuperCollider versions from 3.6 to 3.10. There is no need for any Quarks or Extensions but the two classes HhSounds.sc and HhSdif.sc need to be in SuperCollider’s Extensions folder. For more about these classes see APPENDIX 9.

All my pieces are for eight channels or more. In most instruments I have provided the option to play the piece in stereo. Everything will not work when stereo is chosen, for example some pannings.

The APPENDIX have further information on how to play the instruments.

2.4 SOME ADDITIONAL THOUGHTS REGARDING THE INSTRUMENTS

The question when two different versions of the same piece of music can still be considered to be the same piece is a big question. In pop and rock music a "cover" of another artist’s song can be very different from the original, but we still consider it to be a version of the original. In other cases, a variation may be considered a new piece. When playing my instruments and changing the parameters, the question arises whether a new version or a new piece has been created.

I think that my answer to this question is: My piece dedicated instruments are very limited in what they can do. They all create a distinct sound world. When playing you can change the order of sounds, their durations, their amplitude, their combinations etc, but you can't get out of the sound world they produce. That is why I don't think that you can play another piece than the instrument was made for.

All my instruments use a lot of randomness which means that I don't know exactly what sounds will be produced, but the randomness is quite controlled. I usually only allow values within a certain range so that I know that what comes out will not differ too much each time it is played.

In this portfolio all pieces are of 9.12 or 12.09 duration. This is because I wanted a connection to my mother’s date. When playing my pieces live they can of course be of any length, I would still consider it being the same piece.

A common characteristic of all my current instruments are that musical changes tend to be slow. It takes a while for a new sound to be heard after it has been turned it on. The same goes for turning a sound off. The most obvious reason is that avoiding quick changes makes it easier to create “safe” instruments. I would definitely not want having a sound suddenly played way too loud, or overloading the computer, etc. When coding my instruments and their sound generating functions I always try to check that I will not exhaust the computer’s resources. Such checking can of course be done in many ways, and I may develop instruments with quicker musical changes in the future. For the pieces in this portfolio, I just haven’t had a need for it.

3. MY PIECES

3.1 ABOUT THE DECEMBER 9 RECORDINGS

All of this portfolio is dedicated to my mother. I work with recorded sounds. To make all pieces in this portfolio connected to my mother I have chosen to make recordings on the date that became so important in her life, December 9. As noted above, this created a symbolic connection to her and her life. Most of the December 9 recordings have not been planned in advance. In the first recording from 2013 I had planned to record the Eiffel Tower’s metal structure with contact microphones but that failed as I will describe in chapter 4.3. Instead I simply recorded what was going on around me while standing under the Eiffel Tower. This experience led me to adopt a similar approach to the rest of the recordings, accepting what was possible and what happened, and not trying to plan too much. The recordings done the following years were done in the same manner. While it is not possible to approach this completely without any choices, some time during the day I pick up the recording equipment available, sometimes just an iPhone, and record what is going on. This results in surprising material and gives the recordings a character which I could not have anticipated. I find that this enriches the work.

These unplanned recordings were practical for me in more ways. I'm a bit fatalistic and usually prone to affect my surroundings as little as possible. Even if everything that happens isn't meaningful, it has consequences. Doing the recordings like I did had a big impact in the resulting pieces, but I was very passive. The method can be seen as a way to avoid responsibility - this is what I got (don't blame me), and it also introduces constraints - this is what I have to work with.

The recordings are:

- 2013 - A recording of people and traffic noise from under the Eiffel Tower in Paris, recorded with a ZOOM H6.
- 2014 - Recordings of café noise, a walk on High Street and of a ride with Bus no. 1 in Birmingham, recorded with two iPhones and a laptop.
- 2015 - Leaf blowing gardeners on William Street and a walk on High Street in Birmingham recorded with a ZOOM H6. On the High Street recording one can hear a preacher, street musicians and some music by the band Boney M.
- 2016 - A walk on Centralbron in Stockholm, Sweden recorded with a ZOOM H6.
- 2017 - A concert space in Stockholm, Sweden recorded with an iPhone.

3.2 TITLE, DURATION AND NUMBER OF CHANNELS

NINE STATIONS	12.09	8 channels
UNDER THE EIFFEL TOWER	12.09	8 channels
BUS NO. 1	9.12	8 channels
INUTI	9.12	12 channels
INANNAN	12.09	12 channels
MISKAS	12.09	12 channels

3.3 CHANNEL ROUTING

All pieces in this portfolio share the same channel routing. The pieces that use 8 channels should be routed like this:

```
 1 2
8   3
7   4
 6 5
```

In the pieces that use 12 channels the first 8 channels are routed as in the 8 channel pieces. The additional 4 channels should go to an upper ring routed like this:

```
9  10
12 11
```

3.4 ABOUT TIME

When programming it is easier to treat time as “number of seconds” instead of the perhaps more readable form: “hours.minutes.seconds”. For example it might be easier to read time as 04.32 than 272. To make it easier to convert between the two systems I have included a “number of seconds” to “minutes.seconds” conversion table in APPENDIX 12.

3.5 SPATIALISATION

As soon as I got the possibility to do so (by gaining access to EMS multichannel studios) in 2008 I have only made pieces for a minimum of four speakers. The reason for this is that I want to be surrounded. I have a video work for four video projections and eight channel sound, so the wish to be surrounded or “inside” my own creations is not restricted to music. But, this does not mean that I’m using advanced spatialisation techniques. I don’t. I spread out the sounds I’m using in the number of speakers available. I seldom pan. When there are height differences in the speaker setup I often route the higher frequencies to the ceiling. I find it practical to be able to adapt pieces for different speaker setups and like Benjamin Thigpen describes in his text *Spatialization Without Panning (2009)*, one does not have to pan sounds to create a perception of movement: “’*Ping-pong’* *effects – quick successions of similar sounds in different speakers – can produce the illusion of a moving object*”[9]. I often play small parts of similar sounds in all available speakers. Not at once but with short, often random delays between them. This often sounds as though the sounds move even though they don’t.

3.6 SONOGRAMS

For all pieces in this portfolio there is a picture like Figure 4 below. It is a sonogram exported from AudioSculpt. Before making the sonogram I mix down all audio channels to mono. The image is cropped. There are explaining lines and numbers on the image. These numbers and lines are described in the following text. The images are a bit too small to see properly. The original big images can be viewed in the APPENDIX folder images.

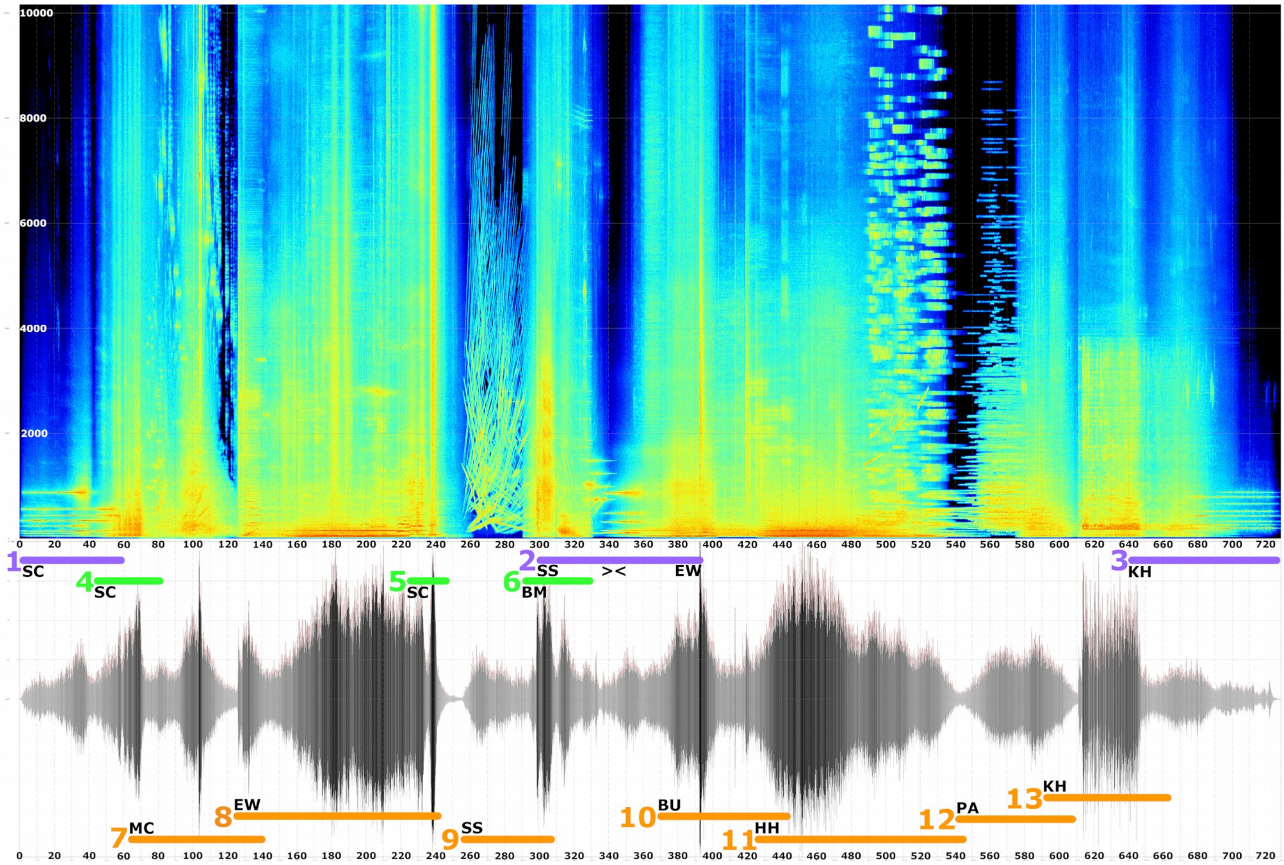


Figure 4: Example sonogram

[9] CEC – e-Contact!, 11-4 – Spatialization Without Panning – Benjamin Thigpen. (Online journal without page numbers.) URL: https://econtact.ca/11_4/thigpen_spatialization.html October 7, 2020

3.7 LIST OF USED SOFTWARE

SuperCollider

- “SuperCollider is a platform for audio synthesis and algorithmic composition, used by musicians, artists, and researchers working with sound. It is free and open source software available for Windows, MacOS, and Linux.” From: https://supercollider.github.io/. August 20 2019.
- SuperCollider is the main sound producing software I use. It has been used in all pieces in this portfolio.

AudioSculpt

- “AudioSculpt is a software for viewing, analysis and processing of sounds. AudioSculpt offers several graphical representations of the sound signal - waveform, spectrum and sonogram - to obtain the desired changes and help the user to select optimal control parameters.” From: http://anasynth.ircam.fr/home/english/software/audiosculpt. August 20 2019.
- AudioSculpt has been used in one way or another in all pieces in this portfolio. Mainly for sound editing and export of analysis SDIF files.

SoundStudio

- A sound editing tool for MacOS X: https://felttip.com/ss/. August 20 2019.
- SoundStudio is the tool I use for cutting sound files.

ProTools

- A Digital Audio Workstation (DAW). https://www.avid.com/pro-tools. August 20 2019.

Paulstretch

- “Paulstretch is a time-stretching of audio designed for extreme stretching”. From: http://www.paulnasca.com/open-source-prhojects#TOC-Paul-s-Extreme-Sound-Stretc. August 20 2019.

Spear

- “Sinusoidal Partial Editing Analysis and Resynthesis”. From: http://www.klingbeil.com/spear/ August 20 2019.
- I used Spear for spectrum manipulations in my piece *UNDER THE EIFFEL TOWER*.

4. COMPOSITIONS

4.1 *NINE STATIONS* I

NINE STATIONS exists in two versions. As a fixed media piece made in 2013, and as an instrument made in 2018. I'll describe both versions below. When comparing the two versions of the piece I prefer the second one and have included *NINE STATIONS* II in this portfolio. (I dislike some of the sounds used in the first version and as follows below, I don't like that the stations in the first version appear in the wrong order.) *NINE STATIONS* I can be found in the APPENDIX folder.

- 2013, fixed media version.
- CHANNELS: 8
- DURATION: 12:09
- TOOLS USED: AudioSculpt, ProTools, SoundStudio and SuperCollider.
- RECORDINGS: All sounds were recorded with a ZOOM H6
- STUDIOS USED: University of Birmingham Electroacoustic studios and EMS in Stockholm.

4.1.1 CONNECTION TO DECEMBER NINE

In this first piece I did not yet have a recording from December 9, so the connection to the date is that I use nine stations, nine frequency scales and the piece duration, 12:09.

4.1.2 IDEA BEHIND THE PIECE

I don't fly. I travel by train. The sounds that make up this piece are recordings of nine railway stations that I have visited on my way between Stockholm, Sweden and Birmingham, UK. (I also made a short trip to Edinburgh.) The stations and the abbreviations I will use hereafter are:

SC = Stockholm Central (Sweden).

SS = Stockholm Södra (Sweden).

MC = Malmö Central (Sweden).

HH = Hamburg Hauptbahnhof (Germany).

KH = Köln Hauptbahnhof (Germany).

BM = Bruxelles Midi (Belgium).

PA = Paris Gare d'Austerlitz (France).

BU = Birmingham University (UK).

EW = Edinburgh Waverley (UK).

My original idea was to:

- Extract a nine frequency scale from each station, i.e. the nine loudest frequencies from a recording with mostly ambient station sounds.
- From each station find characteristic sounds, i.e. sounds that are unique to this station.
- Filter all stations' sounds through these nine frequencies with an RQ that changes over time, from wide where you can hear all of the sound to narrow where the filter starts to self resonate and sound bell like.
- On top of this I wanted to play the characteristic sounds on my instrument (see below).
- I imagined that the finished piece would become a journey, from Stockholm to Birmingham.

I finished the piece and forgot about it. But… when I started to write this text I went back to *NINE STATIONS* - I and examined it. I remembered my intention but when I looked at it I realized that the resulting piece did not adhere to my intention, I decided to make a new version that did. See *NINE STATIONS* - II below.

4.1.3 WORK METHOD

To find the nine loudest frequencies I made functionality to analyse partial analysis files exported from AudioSculpt. The exported analysis is of SDIF (Sound Description Interchange Format) format. I rounded all listed frequencies, counted them and added their amplitude. When I did this I usually got adjacent frequencies, for example 100, 101, 102 etc. This is because the sounds have frequency bands of different “thickness”. To avoid this I clumped the sounds together in frequency bands. I checked if the frequencies were adjacent. If they were I considered them to be the same frequency. When this was finished I could choose the loudest frequency bands, the thickest or a suitable selection with adjusted parameters for loudness or thickness.

The resulting nine frequency scales for each station (in Hertz):

SS: 164, 360, 867, 933, 1734, 1868, 2602, 2802, 3180.

SC: 48, 97, 143, 195, 237, 339, 437, 558, 875.

MC: 99, 195, 294, 495, 743, 988, 1037, 1235, 1480.

HH: 45, 88, 131, 283, 474, 689, 874, 1168, 2100.

KH: 54, 196, 261, 327, 430, 518, 659, 792, 878.

BM: 37, 192, 419, 570, 855, 2609, 3001, 4503, 4601.

PA: 39, 201, 291, 500, 589, 680, 1178, 1767, 1980.

BU: 41, 62, 81, 120, 266, 388, 419, 509, 639.

EW: 50, 100, 116, 149, 316, 382, 480, 847, 872.

4.1.4 DESCRIPTION OF THE PIECE

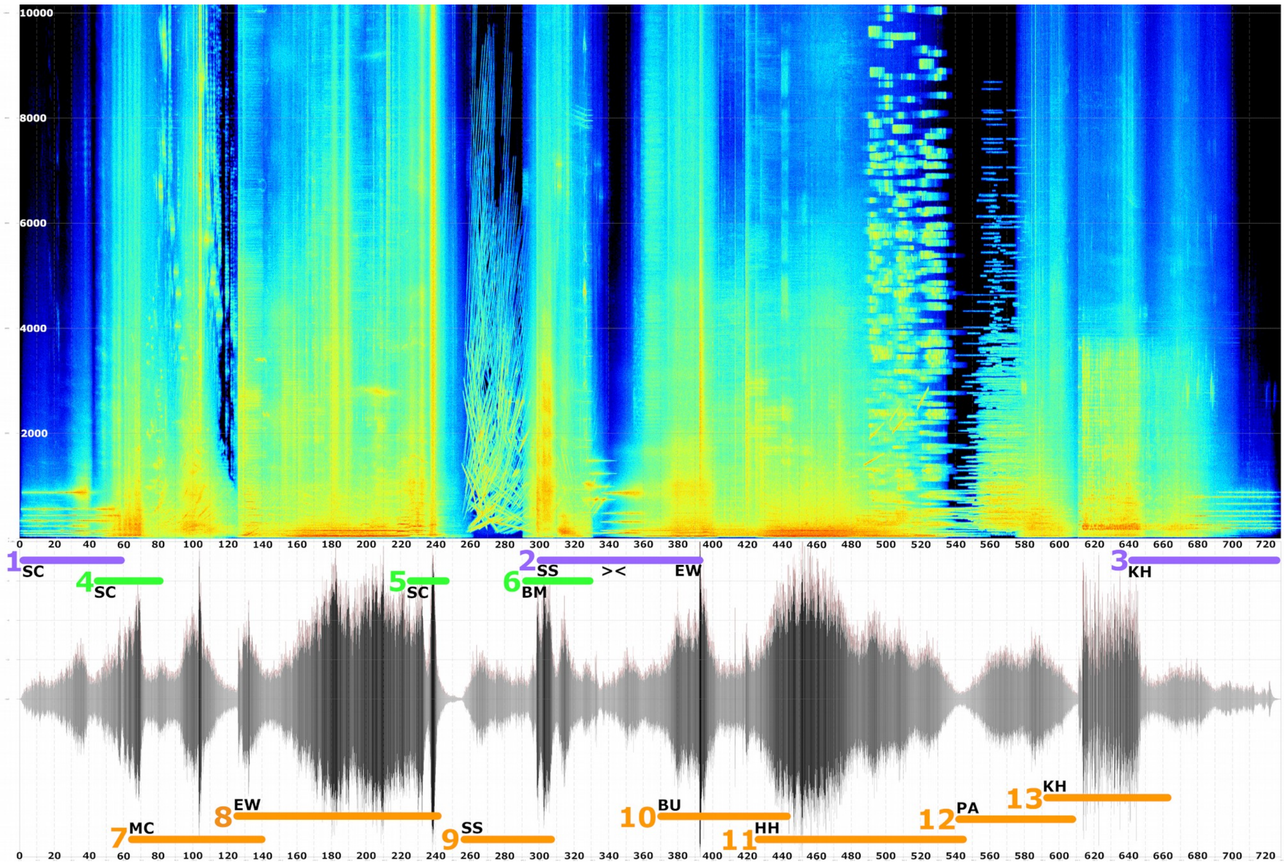


Figure 5: Sonogram of NINE STATIONS - I

Abbreviations used in the “Explanation of the numbers on Figure 5”:

- TW: To wide. From a low RQ value/thin filter to a high RQ value which makes a wider filter.
- TT: To thin. The opposite of TW.
- XF: Cross fade.
- CH: A characteristic sound.

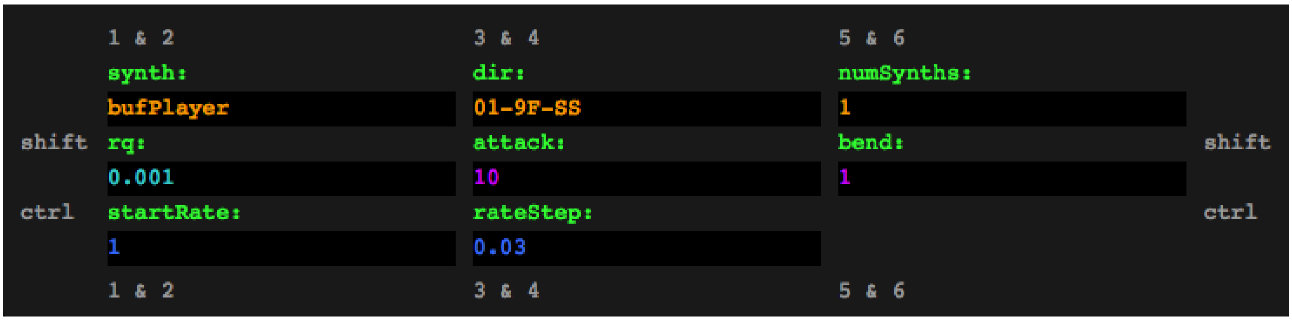
Explanation of the numbers in Figure 5:

- 1: 000-062 Stockholm Central nine frequency scale filtered TW.
- 2: 298-394 Stockholm Södra nine frequency scale filtered TT and crossfaded over to Edinburgh Waverley nine frequency scale TW.

- 3: 640-729 Köln Hauptbahnhof nine frequency scale filtered TT.
- 4: 043-088 A Stockholm Central train passing by.
- 5: 224-248 A Stockholm Central train passing by.
- 6: 290-332 A Bruxelles Midi train passing by.
- 7: 061-142 CH from Malmö Central. Whirring, crackling and whistling train wheels.
- 8: 255-310 CH from Edinburgh Waverly. A low frequency blowing or snorting diesel train.
- 9: 425-547 CH from Stockholm Södra. Moaning brake glissando sounds.
- 10: 540-610 CH from Birmingham University. A low frequency blowing or snorting diesel train standing still.
- 11: 591-665 CH from Hamburg Hauptbahnhop. Fluttering station sounds and whistling wheel sounds.
- 12: 123-243 CH from Paris Austerlitz. Distant sounds of someone playing a piano inside the station.
- 13: 368-457 CH from Köln Hauptbahnhof. Reverberating station announcements.

The journey is not well organised. It goes back and forth between stations in a random order. I was stressed because of a deadline when I made the piece and used my ears to decide where the different sounds should be. This has been changed in the new version of the piece. All sounds were made by playing on my first generic instrument. I recorded the output and finished the piece in ProTools, using ProTools to place my instrument recordings on its timeline.

4.1.5 DESCRIPTION OF THE FIRST GENERIC INSTRUMENT



```
      1 & 2            3 & 4            5 & 6
      synth:           dir:             numSynths:
      bufPlayer        01-9F-SS         1
shift rq:              attack:          bend:           shift
      0.001            10               1
ctrl  startRate:       rateStep:                        ctrl
      1                0.03
      1 & 2            3 & 4            5 & 6
```

Figure 6: The first generic instrument used in NINE STATIONS - I *and UNDER THE EIFFEL TOWER* - I

This is a simple instrument that is not dedicated to a certain piece. You play it by using the computer keyboard similar to a keyboard instrument. I used it in the next piece *UNDER THE EIFFEL TOWER* - I also. You can choose a synth/play method, a sound directory, how many simultaneous synths will be played. If you have chosen a filter synth you can control the RQ value. If you have chosen a fade-bend synth you can control the attack and bend amounts. You can also retune the keyboard by changing the startRate and rateStep. This instrument can not be used to play a tonal piece.

For details about this instrument, see APPENDIX 1.

4.2 *NINE STATIONS* II

- 2018, recorded instrument version.
- CHANNELS: 12
- DURATION: 12:09
- TOOLS USED: AudioSculpt and SuperCollider.
- RECORDINGS: The same original recordings as in *NINE STATIONS* - I were used but they were edited from scratch again.
- STUDIOS USED: EMS in Stockholm.

4.2.1 WORK METHOD

When turning NS into an instrument I had to decide what I wanted to be able to control. The RQ altering stations were obvious as was the characteristics. When starting the *NINE STATIONS* - II instrument it begins to play a chosen station. The RQ value changes automatically if you don’t freeze it, i.e. stops it on the value where it currently is. On top of this you can play the currently chosen stations’ characteristic sounds. I disliked some of the characteristic sounds that I used in *NINE STATIONS* - I, for example the glissando brakes so I removed them. I also found new sounds, for example I swapped the PA pianist for the sounds of an ongoing construction work in the same station.

When working with the nine frequency scales I found some of them irritating after listening to them for a while. Therefore I wanted to be able to control which frequencies to use simultaneously. I also added the possibility to make the frequencies “tonal”, i.e. to move them to the nearest frequency in a normal 12-ET scale. With this addition I was able to play real tonal chords if I wanted to.

When I began to work on this second version I went back to my original recordings and edited them from scratch. Therefore I used other parts of the recordings and therefore I got other frequencies.

The resulting nine frequency scales for each station (in Hertz):

SS: 931, 961, 996, 999, 1026, 1163, 1180, 1456, 1736.

SC: 41, 235, 1109, 1147, 1178, 1211, 1256, 1329, 1357.

MC: 98, 140, 196, 490, 740, 988, 1042, 1234, 2966.

HH: 40, 280, 1071, 1191, 1246, 1371, 1387, 1452, 1553.

KH: 63, 204, 1069, 1096, 1128, 1147, 1175, 1329, 2665.

BM: 50, 212, 1115, 1161, 1194, 1239, 1356, 1426, 4477.

PA: 75, 118, 298, 1229, 1263, 1311, 1447, 1683, 2012.

BU: 44, 94, 204, 263, 365, 428, 659, 1002, 5129.

EW: 41, 80, 102, 159, 1282, 1292, 1315, 1401, 1445.

4.2.2 DESCRIPTION OF THE PIECE

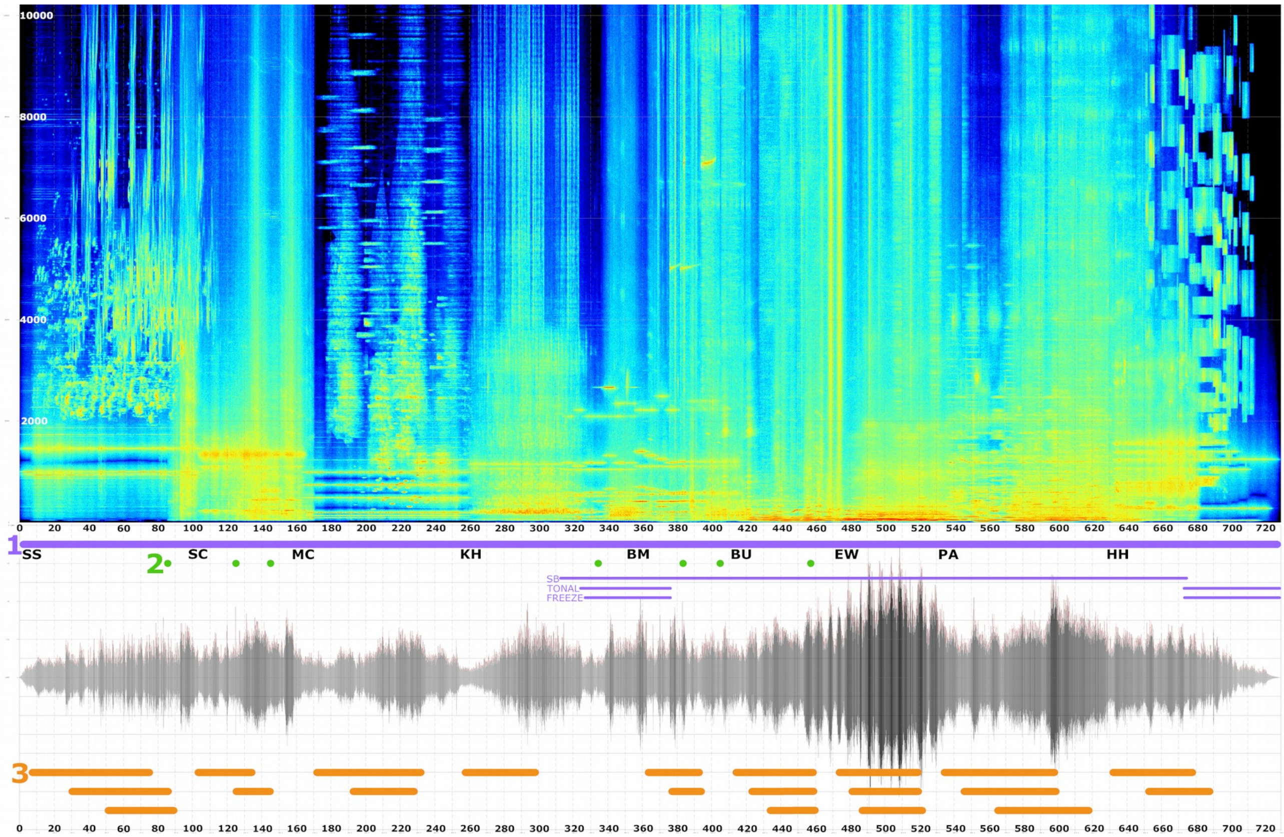


Figure 7: Sonogram of NINE STATIONS - II

Explanation to the numbers in Figure 7:

- 1: The stations’ nine frequency scales run throughout the piece but usually not with all nine frequencies at the same time. The RQ and amplitude values go from 0.005 (RQ) and 8 (amplitude), to 0.1 (RQ) and 0.8 (amplitude) and back again. It takes 60 seconds to go from the smallest value to the largest value, and another 60 seconds to come back to the beginning again. See Table 1: Time, station and frequency list.
- 2: Passing trains. Times and stations:
  - 086: Stockholm Södra.
  - 125: Stockholm Central.
  - 145: Stockholm Central.

  - 334: Köln Hauptbahnhof.
  - 382: Bruxelles Midi.
  - 405: Bruxelles Midi.
  - 459: Birmingham University.
- 3: Characteristic sounds. Stations, times and types. See Table 2: *NINE STATIONS* - II - Characteristic sounds. Stations, times and types.

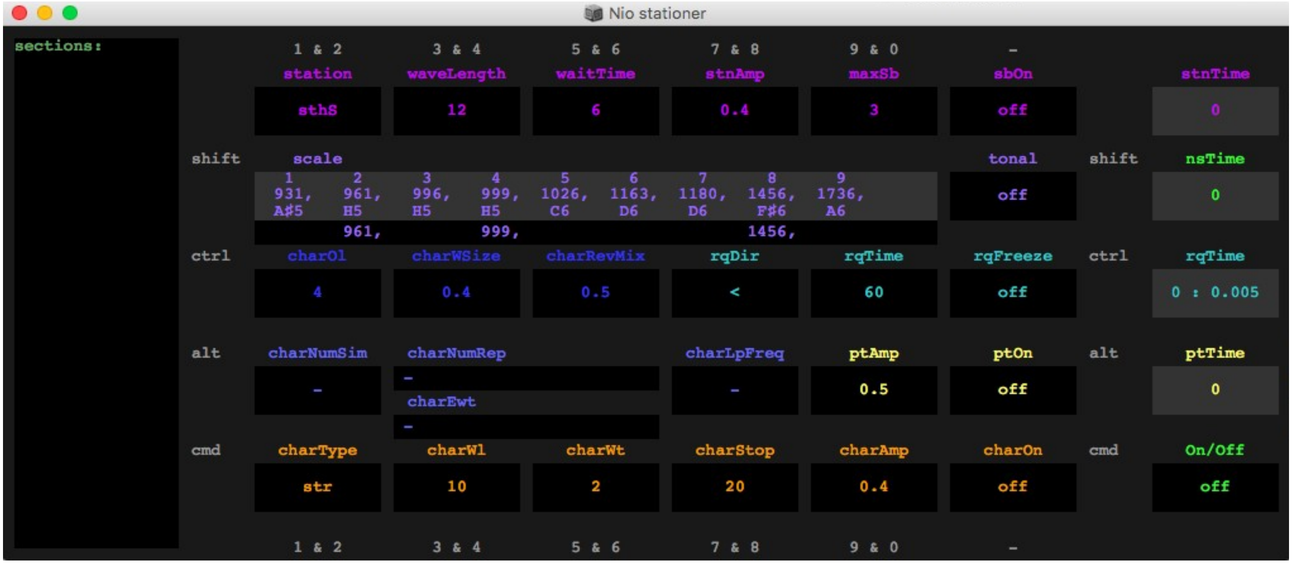
Table 1: NINE STATIONS - II - *Time, station and frequency list.*

Seconds	**Station**	**Used frequencies**						
000	SS	961	999	1456				
098	SC	41	235	1109	1329	1357		
158	MC	98	140	196	490	740	988	
255	KH	63	204	1069		1147		
312		63	204	1069	1128			
317		63	204	1069		1147		
322		63	204	1069			1175	
328		63	204	1069				1329
334		63	204	1069			1175	
340		63	204	1069		1147		
346		63	204	1069	1128			
352	BM	50	212	1115				1426
358		50	212	1115			1356	
364		50	212	1115		1239		
370		50	212	1115	1194			
412	BU	94	204	263	365	428		
472	EW	41	80	102	159			
531	PA	75	298	1311	2012			
629	HH	40	280	1071	1246	1387	1553	
696		40	280	1071	1246	1387		
702		40	280	1071	1246			
708		40	280		1246			
714			280		1246			
720					1246			
729								

Table 2: NINE STATIONS - II - *Characteristic sounds. Times, stations and types.*

Seconds	**Station**	**Description**
006-077	SS	Time stretched high pitched moaning sounds.
028-087	SS	Down pitched very high frequency bird song like sounds.
049-092	SS	The same high pitched sounds as above played through an echo effect.
102-136	SC	Time stretched sound of a freight train passing by.
124-146	SC	The same sound as above but unmanipulated.
170-234	MC	Time stretched sounds of train brakes.
191-230	MC	Time stretched sounds with the short melody that precedes station announcements.
256-300	KH	Station announcements.
362-395	BM	Platform sounds. Blowing engines and luggage drawn along the platform.
376-396	BM	Sounds from blowing engines played through a percussive effect.
413-461	BU	Low pitched sounds.
422-462	BU	Time stretched sounds of blowing trains.
433-463	BU	Sounds of diesel trains.
473-522	EW	Low pitched sounds.
479-523	EW	Sounds from still standing diesel trains.
485-524	EW	Sounds from blowing engines played through a percussive effect.
533-600	PA	Filtered sounds from workers mending something in the station.
544-601	PA	The same type of sounds as above played through a percussive effect.
564-620	PA	Sounds from workers mending something in the station.
630-680	HH	Time stretched sounds from whirring wheels on tracks.
651-690	HH	The same sound as above played through an echo effect.1

4.2.3 DESCRIPTION OF THE INSTRUMENT



```
Nio stationer
sections:
              1 & 2        3 & 4        5 & 6        7 & 8        9 & 0        -
              station      waveLength   waitTime     stnAmp       maxSb        sbOn                stnTime
              sthS         12           6            0.4          3            off                 0
shift         scale                                                            tonal       shift   nsTime
              1      2      3      4      5      6      7      8      9
              931,   961,   996,   999,   1026,  1163,  1180,  1456,  1736,   off                 0
              A#5    H5     H5     H5     C6     D6     D6     F#6    A6
                     961,          999,                        1456,
ctrl          charOl       charWSize    charRevMix   rqDir        rqTime       rqFreeze    ctrl    rqTime
              4            0.4          0.5          <            60           off                 0 : 0.005
alt           charNumSim   charNumRep                charLpFreq   ptAmp        ptOn        alt     ptTime
              -            -                         -            0.5          off                 0
                           charEwt
                           -
cmd           charType     charWl       charWt       charStop     charAmp      charOn      cmd     On/Off
              str          10           2            20           0.4          off                 off
              1 & 2        3 & 4        5 & 6        7 & 8        9 & 0        -
```

Figure 8: NINE STATIONS - II *instrument GUI*

This instrument is dedicated to *NINE STATIONS* - II. In the top cerise row you control which stations’ sounds should be played, their amplitude and a few other things. In the lilac row below you control parameters that have to do with the nine frequency scales. The blue and orange buttons control the playing of the characteristic sounds. The turquoise buttons control RQ related parameters. The yellow buttons control the sounds of passing trains. The yellow buttons can only be used if the chosen station has sounds of passing trains.

For a more detailed description of the instrument and how to play it, see APPENDIX 3.

4.2.4 AFTERTHOUGHTS

NINE STATIONS - II was the second instrument that is dedicated to a single piece that I made. There are lots of things that I would like to rewrite, and I probably will in the future, but I learnt a lot by making it.

I have only actually played it live a few times. I find it difficult to play, but this difficulty would probably go away with more practice. As I have so many instruments I forget which control is where and have to look for it. This takes time. When I played it live I had a hand written note telling me exactly what to do and when, and I practiced a lot before the concert. But, compared to when I used to play in rock bands, it was not a very musical experience, this following of instructions and finding the right keys to press. I couldn't really listen to what I produced. To become a nice experience, either my instruments must become simpler, or I have to practice much more.

About the difference between the two versions. As I mentioned before, I disliked some sounds that I used in *NINE STATIONS* - I so I replaced them with new sounds in *NINE STATIONS* - II. (Not completely new, only new parts of the original ZOOM H6-recordings.) I also made it possible to control how many of the nine frequencies to listen to at the same time. In *NINE STATIONS* - I the stations are passed in an unorganised order, in *NINE STATIONS* - II they are passed in travel order. *NINE STATIONS* - II use twelve channels with four speakers in the ceiling instead of 8 on the floor and I miss the hight dimension when I listen to *NINE STATIONS* - I. I prefer the non fixed version and like that I can easily change timings and amplitudes and that I can avoid certain sounds or play methods if I get tired of them. Of course I can as easily add new play methods. This flexible semi unfixed way suits me well. I call it semi because the *NINE STATIONS* - II instrument can not play another piece. With the supplied sounds, play methods, the nine frequency scales and the characteristic sounds it will – in my opinion - always be a version of *NINE STATIONS* - II.

Was the project successful? I wanted to extract nine frequency scales from noisy sounds. I had an idea that if I took the nine loudest frequencies from completely atonal sounds and constructed a scale based on these frequencies, I could produce something that sounded interesting. I wanted to go from the actual noisy sound to almost pure sine tones from the nine frequencies and back. I also wanted to “tune” the characteristic sounds according to the nine frequencies. I imagined that that the two different sound layers would sound connected. Was I successful? Yes and no. I can hear the sounds grow out of the thin bell like sounds but in my opinion, the characteristic sounds don't sound connected to the nine frequencies. That's probably because they often aren't. I tried to “tune” every stations’ characteristic sounds according to the stations’ nine frequency scale. Many of the sounds simply sounded “bad” when transposed. I kept those that sounded “good”, but the transpositions don't make the two layers sound connected.

4.3 *UNDER THE EIFFEL TOWER* I

UNDER THE EIFFEL TOWER exists in two versions. As a fixed media piece made in 2014, and as an instrument made in 2019. I’ll describe both versions below. When comparing the two versions of the piece I prefer the first one and have included *UNDER THE EIFFEL TOWER* I in this portfolio. (I find version number two a bit indistinct and a bit too even regarding dynamics.) *UNDER THE EIFFEL TOWER* II can be found in the APPENDIX folder.

- 2014, fixed media version.
- CHANNELS: 8 (The original version is for 40 channels.)
- DURATION: 12:09.
- TOOLS USED: Audacity, AudioSculpt, ProTools, SoundStudio, Spear and SuperCollider.
- RECORDINGS: All sounds were recorded with a ZOOM H6.
- STUDIO USED: Atelier 4 and Kubus Klangdom at ZKM, Karlsruhe, Germany.

4.3.1 CONNECTION TO DECEMBER NINE

On December 9, 2013 I was in Paris to celebrate my birthday. I went to the Eiffel Tower – I had planned to record the actual tower with contact microphones, but the queue was too long. Instead I stood under the tower for a while and recorded what went on around me. Bird song, people talking in different languages, construction work, an emergency vehicle passing by. The day after, December 10 I went back and recorded the tower as planned. It was very disappointing. I thought that I would get interesting sounds from the metal moving. Instead I got a mixture of different French radio broadcasts. The tower is a radio mast.

In this piece the connection to December 9 is that the field recording was made on the actual date and the piece's length 12:09. In the finished piece, you never hear the actual field recording but the recording 'played' by a grand piano, and stretched (Paulstretch) in time to 729 seconds. (The original recording is 71 seconds long). The field recording's sonogram (as an image) has been used as a filter for the grand piano.

4.3.2 IDEA BEHIND THE PIECE

In February 2013 I was invited to ZKM in Karlsruhe, Germany for a one month residency. (For me, one month is a very, very short time for making a piece.) The residency ended with a concert in ZKM’s Kubus Klangdome. The Kubus has 43 loudspeakers placed like this:

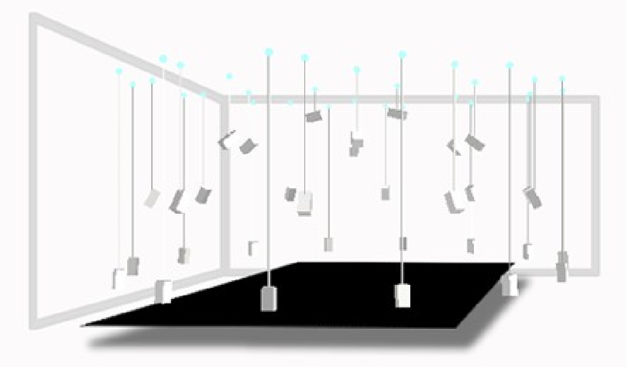


Figure 9: Kubus

I had these plans for the piece:

- To use my recording from under the Eiffel Tower.
- To use piano sounds as sound source.
- To manipulate the piano’s spectrum by turning them upside down and inside out.

4.3.3 WORK METHOD

Before traveling to Karlsruhe I recorded a Yamaha grand piano like this:

- All 88 keys, one at a time, hitting the key hard and letting the tone ring out but without using the sustain pedal.
- Clusters.

- Hammer and key noise.
- Many variants of pressing the keys B, E, D, and E♭ without making a sound, and then hit another B, in order to let the already pressed keys ring with the B[10].

While editing the sounds I transposed all recorded keys/notes to the nearest B. This was done in order to enable playing with my own instrument tuning. I made two sonograms of the field recording in AudioSculpt, using both channels of the stereo recording. I exported the sonograms as images and used these images as filters.

I prepared four files with sounds from the grand piano. They contain:

1. All recorded B-E-D-E♭sounds.
2. All original (not transposed) B keys.
3. All clusters.
4. All original B keys transposed down an octave.

I took a short snippet from each file and granulated it in order to get 71 seconds long, more static sounds. These static sounds were then filtered by the field recording's sonogram, and stretched in time up to 729 seconds (the total duration of the piece).

The spectrum manipulations were done in AudioSculpt and in Spear like this:

To turn the spectrum upside down I took the recorded keys/tones and made sonograms of them. I exported the sonogram as an image. I opened the image in an image manipulation program and simply turned the image upside down. Doing so makes most of the interesting parts of the sound to go up to frequencies higher than humans can hear, so I had to lower it manually by dragging the image down into the audible frequency domain. To turn the image back into sound I filtered a cluster sound file with the image.

[10] In Swedish, as in German, the letter H is used instead of B. H, E, D and E♭ (pronounced as “S” in Swedish) are letters from my name.

UNDER THE EIFFEL TOWER - I spectrum manipulations

Sonogram of the lowest B

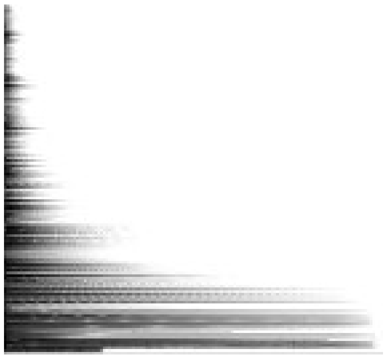


Figure 10: Spectrum manipulations

Upside down

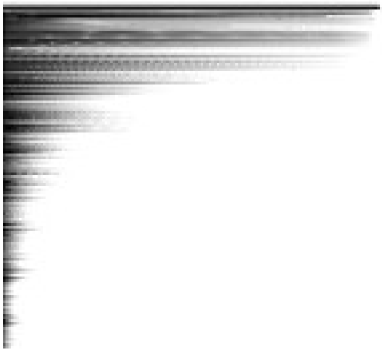


Figure 11: Spectrum manipulations

The upside down sounds were made by filtering the already prepared cluster file. The reason for this is that the original sound file has no frequencies in the areas where the upside down image needs them.

To turn the spectrum inside out was trickier. Again I used exported sonograms from Audiosculpt, but now I tried many different ways of inverting the image. One of them look like this:

UNDER THE EIFFEL TOWER - I spectrum manipulations

Sonogram of lowest A transposed to B

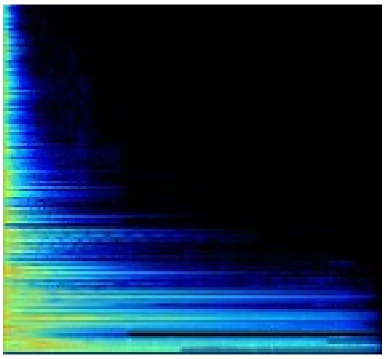


Figure 12: Spectrum manipulations

Inverted colours

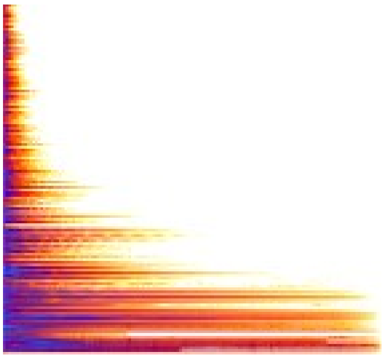


Figure 13: Spectrum manipulations

I also used Spear. In Spear one can “Select partials below threshold” and give a value in dB. I chose for instance -60 dB, inverted the selection and removed all other partials and saved the file. I repeated this process, next time using -50 dB. Later, when I had all partials saved in different files I put it all together again, adjusting each files amplitude and voila! A piano sound turned inside out.

4.3.4 DESCRIPTION OF THE PIECE

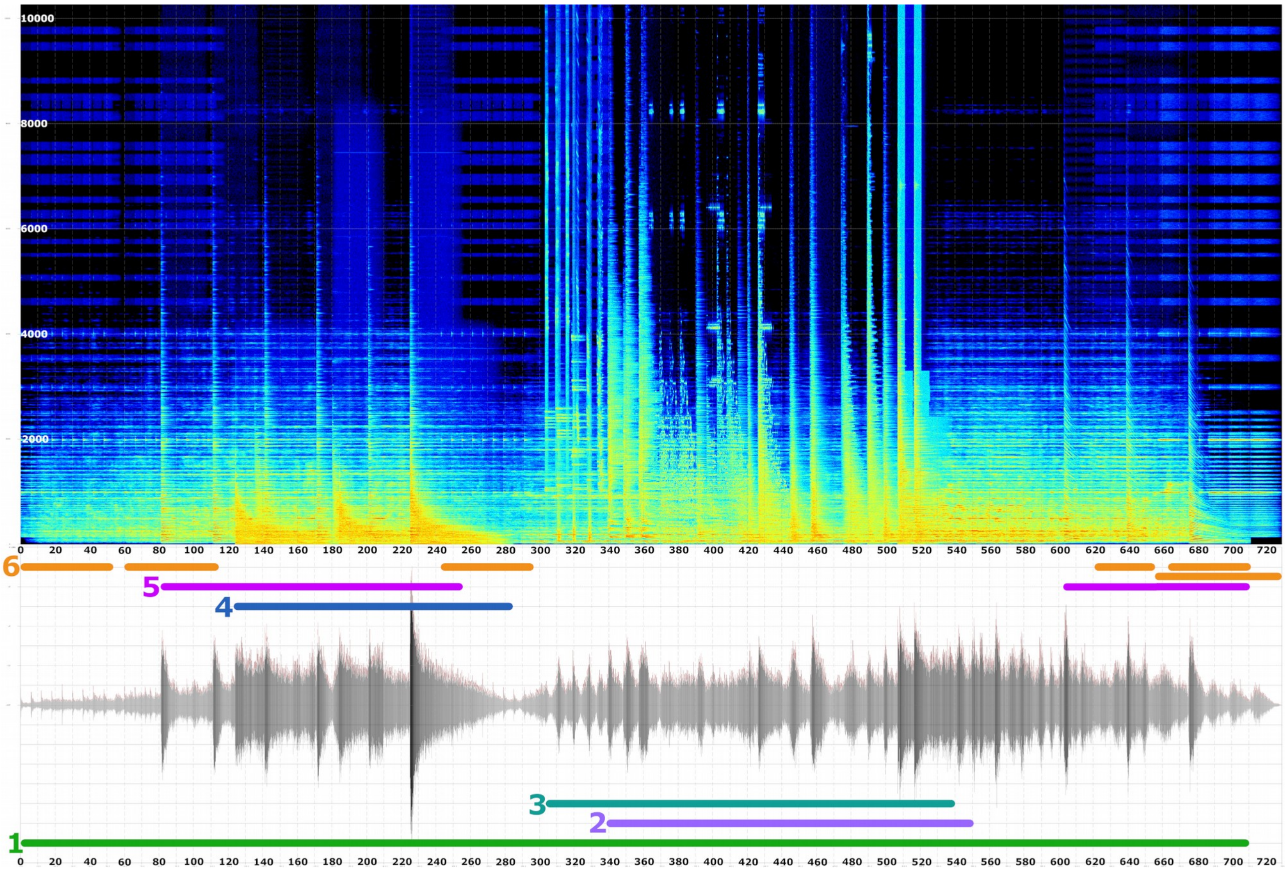


Figure 14: Sonogram of UNDER THE EIFFEL TOWER - I

Explanation to the numbers in Figure 14:

1. The field recording played by the grand piano as described above.
2. A section with the spectrum manipulated sounds played on my instrument.
3. A section of falling piano sound clouds, played on my instrument, starting from the highest octave and ending with the lowest. The hammer sounds were used as a “tail”, they follow the piano sounds.

4. A section of low inverted B sounds.
5. A low frequency B repeating approximately every 30th second. Towards the end of the piece the B becomes a downward glissando.
6. A high frequency B-E-D-E♭-chord repeating every sixth second.

All sounds were made by playing on my instrument. I recorded the output and finished the piece in ProTools.

UNDER THE EIFFEL TOWER - I uses the same instrument as *NINE STATIONS* - I but with the sounds described above.

4.4 *UNDER THE EIFFEL TOWER* II

(UNDER THE EIFFEL TOWER II is not included in this portfolio but can be found in the APPENDIX folder.)

- 2019, recorded instrument version.
- CHANNELS: 12
- DURATION: 12:09.
- TOOLS USED: SuperCollider.
- RECORDINGS: The same recordings and the same spectrum manipulations as in *UNDER THE EIFFEL TOWER* - I.
- STUDIO USED: EMS in Stockholm.

4.4.1 WORK METHOD

Why make an instrument version? Because by now, after making *NINE STATIONS* - II I decided that I wanted all of my portfolio pieces to be instruments. There were three reasons for this. I thought that if I did this it would make my portfolio more coherent. The other reason is that I'm fond of problem solving and I wanted to see if I *could* do it . The third reason is that I wanted to stop using DAWs. Also, since I had so little time to make *UNDER THE EIFFEL TOWER* - I I wasn't entirely happy with the piece. I did like the repeating B E D E♭ sounds and the Paulstretched background file played by the grand piano, but I thought that all the crashing spectrum manipulated sounds got a bit tiring after a while. So, when planning the instrument I wanted to:

- Play the background file in any number of channels and I wanted to be able to control which of the files were played simultaneously. I added the possibility to play the original time stretched field recording (not played by the grand piano).
- Play the repeating B E D E♭ and the low B sounds at any chosen time interval.

- Play the cascading piano sounds with noise tails.
- Play – as before – the spectrum manipulated sounds from the computer keyboard.
- Play the December 9 recording from a partial analysis of the file. (For a description of this, see chapter 2.7 *INUTI*)

4.4.2 DESCRIPTION OF THE PIECE

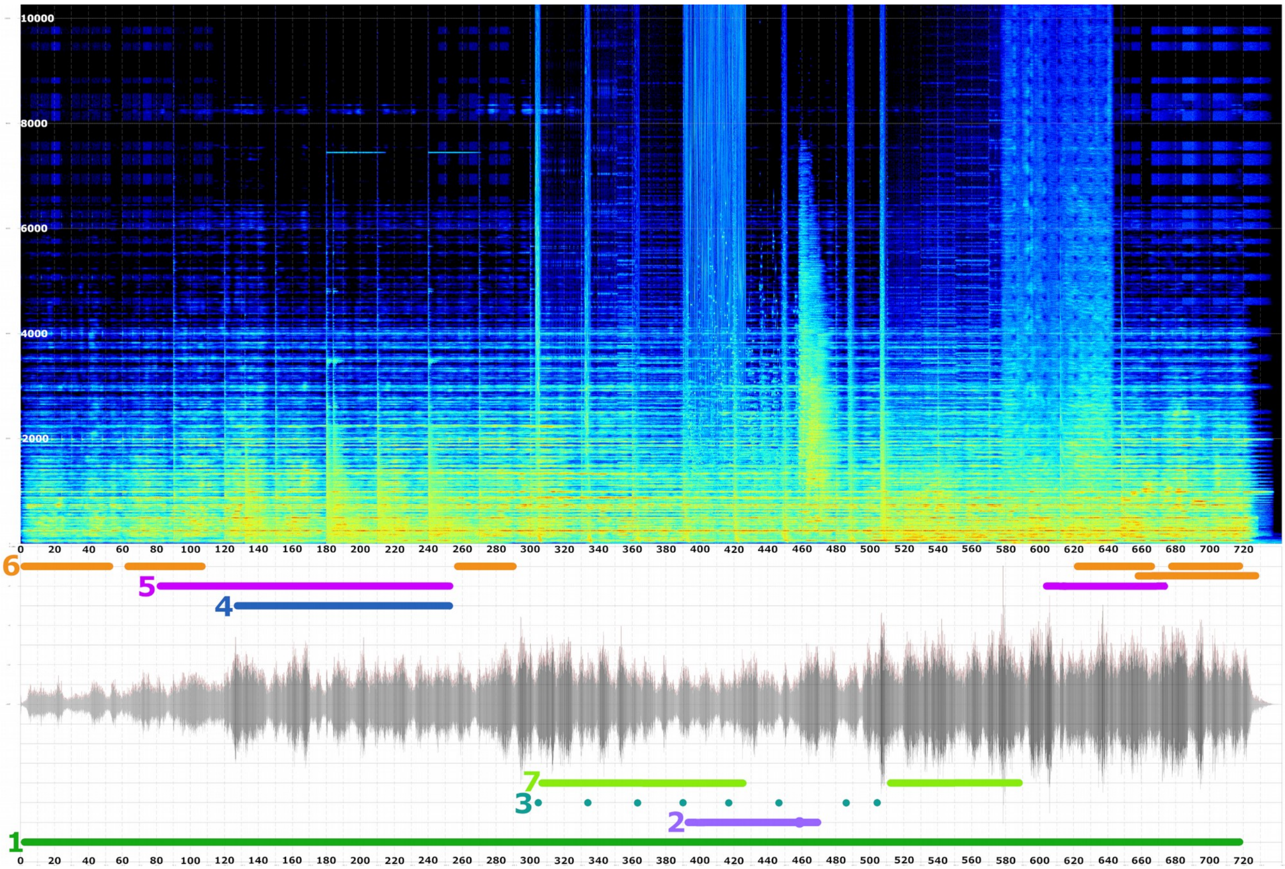


Figure 15: Sonogram of UNDER THE EIFFEL TOWER - II

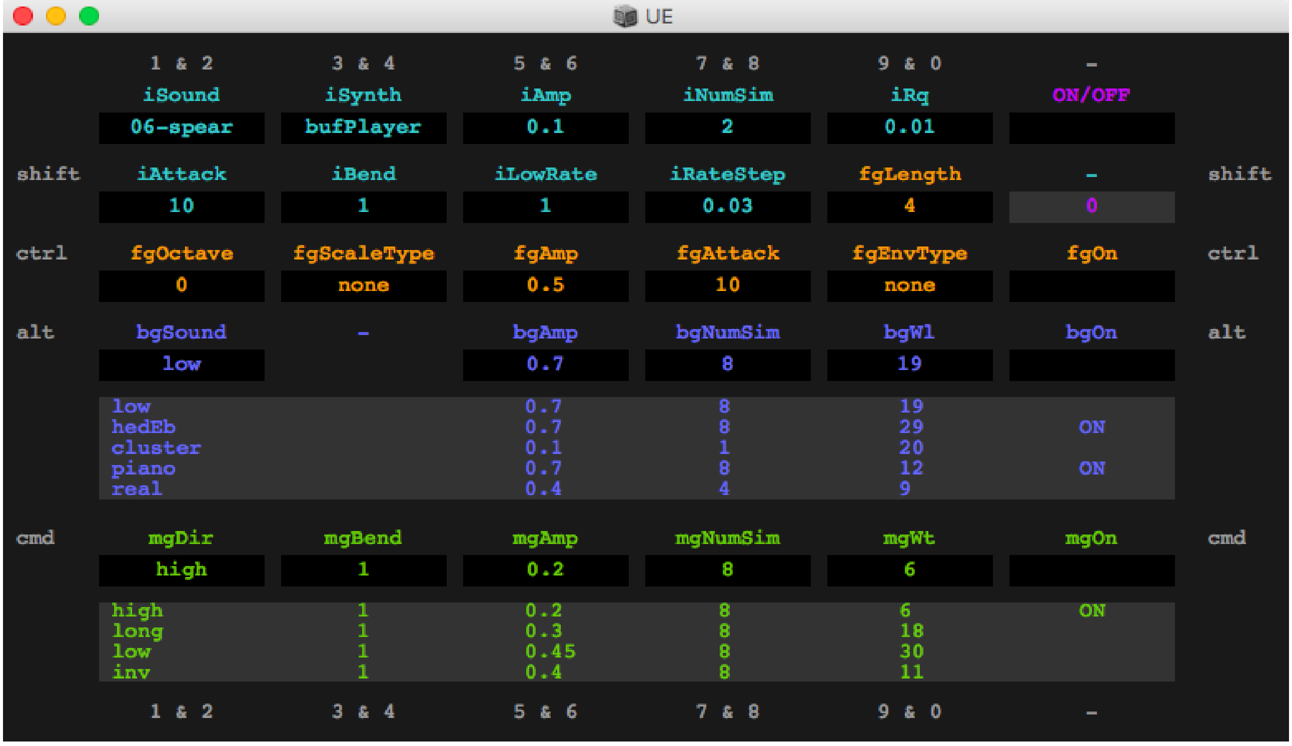
Explanation to the numbers in Figure 15:

1. The field recording played by the grand piano.
2. A section with spectrum manipulated sounds.
3. A programmed version imitating me playing number 3 (as described in number 3 in Description of *UNDER THE EIFFEL TOWER* - I) in my description of *UNDER THE*

EIFFEL TOWER - I. The dots in the image above shows where the different octaves can be heard. It starts on the highest octave and ends with the lowest.

4. A section of low inverted B sounds.
5. A low frequency B repeating approximately every 30th second. Towards the end of the piece the B becomes a downward glissando.
6. A high frequency B-E-D-E♭-chord repeating every sixth second.
7. This part is new and didn't exist in *UNDER THE EIFFEL TOWER* - I. It plays a partial analysis of the field recording.

4.4.3 DESCRIPTION OF THE INSTRUMENT



UE

	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	
	iSound	iSynth	iAmp	iNumSim	iRq	ON/OFF	
	06-spear	bufPlayer	0.1	2	0.01		
shift	iAttack	iBend	iLowRate	iRateStep	fgLength	-	shift
	10	1	1	0.03	4	0	
ctrl	fgOctave	fgScaleType	fgAmp	fgAttack	fgEnvType	fgOn	ctrl
	0	none	0.5	10	none		
alt	bgSound	-	bgAmp	bgNumSim	bgWl	bgOn	alt
	low		0.7	8	19		
	low		0.7	8	19		
	hedEb		0.7	8	29	ON	
	cluster		0.1	1	20		
	piano		0.7	8	12	ON	
	real		0.4	4	9		
cmd	mgDir	mgBend	mgAmp	mgNumSim	mgWt	mgOn	cmd
	high	1	0.2	8	6		
	high	1	0.2	8	6	ON	
	long	1	0.3	8	18		
	low	1	0.45	8	30		
	inv	1	0.4	8	11		
	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	

Figure 16: UNDER THE EIFFEL TOWER - II *instrument GUI*

Description of the instrument (Figure 16) from top left.

Turquoise buttons starting with the letter i (as in Instrument) control the playing on the computer keyboard. You can control parameters such as which sounds to use, which play method and how much amplitude. Orange buttons starting with letters fg (foreground) control the playing of the partial analysis. The five blue buttons with names starting with the letters bg (background) control the playing of the December 9 files that were also used in *UNDER THE EIFFEL TOWER* - I. The six green buttons with names starting with the letters mg (middle ground) control the recurring B E D E♭ and low B sounds.

For a more detailed explanation of the instrument see APPENDIX 4.

4.4.4 AFTERTHOUGHTS

The biggest difference between the two versions of *UNDER THE EIFFEL TOWER* is timing which I had more exact control over in the first DAW version. Another difference is that I removed some of the more irritating spectrum manipulated crash sounds in the second version. I made the instrument so that it would be able to play the first version. If I had made it like an instrument from the beginning I'm sure that it would have become different. This is the piece that I had to finish in only one month and I don't usually make pieces that fast. So, *UNDER THE EIFFEL TOWER* - I is a bit different from the rest. There was only time to realise the plan I had when I came to Karlsruhe, not to experiment.

In the second version I had problems with recreating the high frequency falling piano sound clouds. In the instrument they are programmed, in *UNDER THE EIFFEL TOWER* - I they were played on the computer keyboard and later edited in ProTools.

This is the only submitted instrument where I have kept the possibility to play on the computer keyboard. Therefore it is possible to make a piece that is very different from the submitted piece. In the submitted version of *UNDER THE EIFFEL TOWER* - II I tried to make a piece that was very similar to *UNDER THE EIFFEL TOWER* - I. The instrument was made to be able to recreate the original piece.

The new part where the grand piano plays a partial analysis of the field recording is not very good. I added it after I had done INUTI, but it doesn’t really work in UNDER THE EIFFEL TOWER - II. It is difficult to hear that the two layers, no 1 and 7 in Figure 16, play the same thing, but they do and they are synchronized where ever I choose to start playing the partial analysis data.

4.5 *BUS NO. 1* I

BUS NO. 1 exists in two versions. As a fixed media piece made in 2015, and as an instrument made in 2019. I'll describe both versions below. When comparing the two versions of the piece I prefer the first one and have included *BUS NO.1* I in this portfolio. (In the first version I like the tonality switch, approximately at 270 seconds (4 minutes and 30 seconds) into the piece. This switch is not as clear in version number two.) *BUS NO.1* II can be found in the APPENDIX folder.

- 2015, fixed media version.
- CHANNELS: 8
- DURATION: 9:12.
- TOOLS USED: AudioSculpt, ProTools, SoundStudio and SuperCollider.
- RECORDINGS: The bus and café recordings were made with two iPhones and a laptop. The laundry rack was recorded with a ZOOM H6.
- STUDIOS USED: University of Birmingham Electroacoustic studios and EMS in Stockholm.

4.5.1 CONNECTION TO DECEMBER NINE

On December 9, 2015 I had worked all day at the MAC, the Midlands Art Center in Birmingham, UK. I was working on a dance performance together with two dancers. I recorded the MAC's café during the day and on my way back I and one of the dancers recorded parts of a bus ride, going from the MAC towards Five Ways. We used two iPhones and a laptop, so I got a bad quality quad recording. This piece took a long time to make as I found it hard to make the bus recording sound interesting. In the end, I remembered what Justin Connolly taught me about serialism back in 1996 and took a serial approach.

4.5.2 IDEA BEHIND THE PIECE

The idea was simply to use the December 9 bus sounds. I also had a longing for tonality. After working atonally for many years I really wanted to be able to get some tonality out of my sounds occasionally. By tonality I mean, for example, to go from noise to pitch, from random overtones to natural overtones, from random pitches to pitches organised according to a chosen scale, e.g. A minor.

4.5.3 WORK METHOD

I used this note series (from my name: HElEnE mAriAnnE HEDSunD[11]):

B E E E A A E B E D E♭ D

and these durations:

15, 21, 21, 15, 11, 15, 17, 7, 17, 11, 15, 15 (180 seconds)

I did sixteen bus tracks. The tracks are made up of filtering the bus with a band pass filter (BPF) with different RQ values. This is the data for one out of sixteen simultaneous bus tracks.

Table 3: *BUS NO. 1* - I series example

Note	Duration	RQ	Amplitude
H	15	0.9	0.8
E	21	0.8	0.95
E	21	0.007	5
E	15	0.6	1.1
A	11	0.5	1.2
A	15	0.4	1.3
E	17	0.3	2.0
H	7	0.002	3

[11] In Swedish, as in German, the letter H is used instead of B. H, E, D and E♭ (pronounced as “S” in Swedish) are letters from my name.

E	17	0.001	7
D	11	0.09	1.7
Eb	15	0.008	5
D	15	0.007	3

The next track uses the same series but starts on value number two, so the note series becomes E E E A A E B E D E♭ D B. In table 3 above, for the second E the RQ is 0.007 instead of the expected 0.7. This is because of the bumps. The bus ride was very bumpy and caused the recording to distort. The extra narrow RQ is used to hide the distortion.

The other sounds in this piece come from a recording of my own laundry rack. It makes a screeching sound when a wet sheet is drawn over it. I separated the rack sound from the sheet sound in AudioSculpt. Both of these sounds are used with the same filter series as the bus sounds. On top of this I play laundry rack sounds on my instrument described below. The café sounds are played as they are without modification.

4.5.4 DESCRIPTION OF THE PIECE

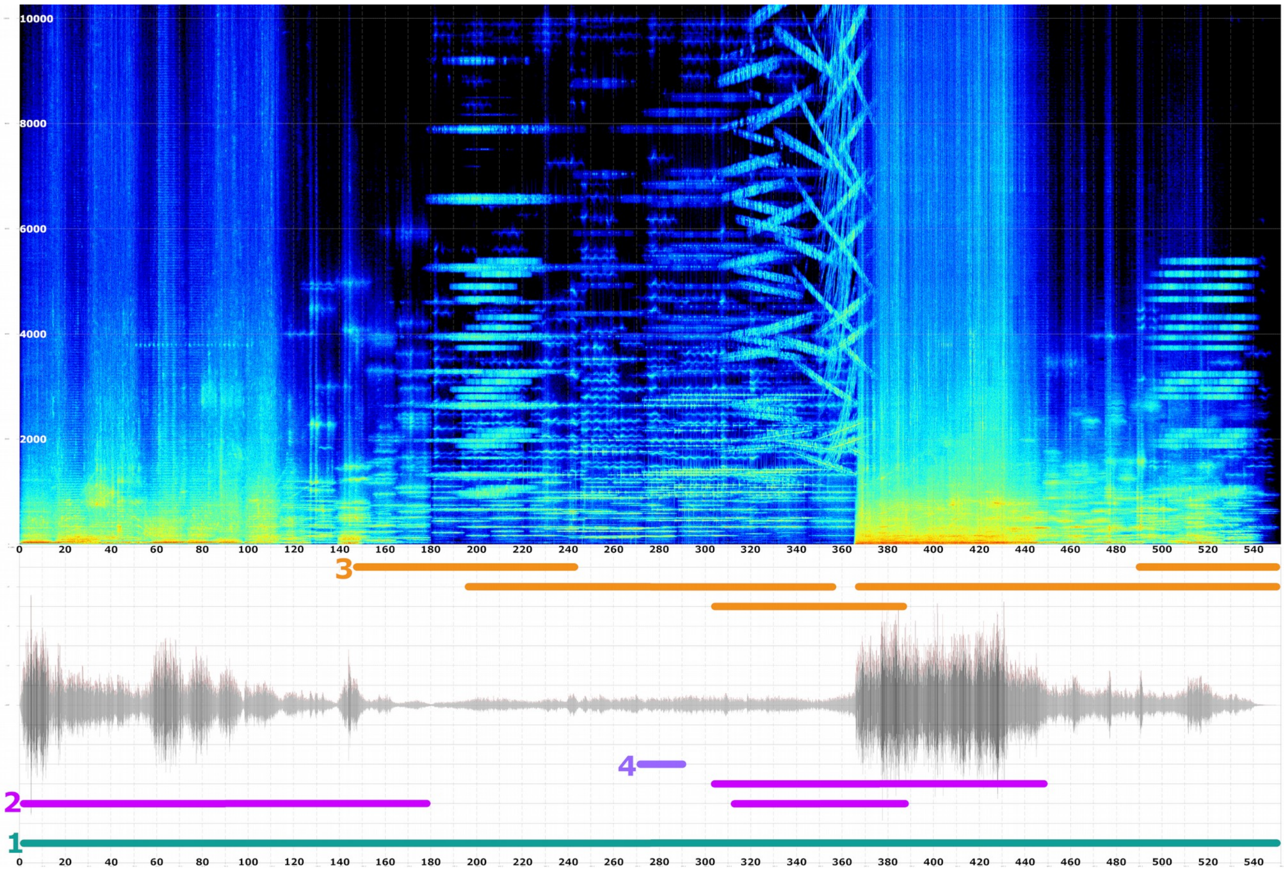


Figure 17: Sonogram of BUS NO. 1 - I

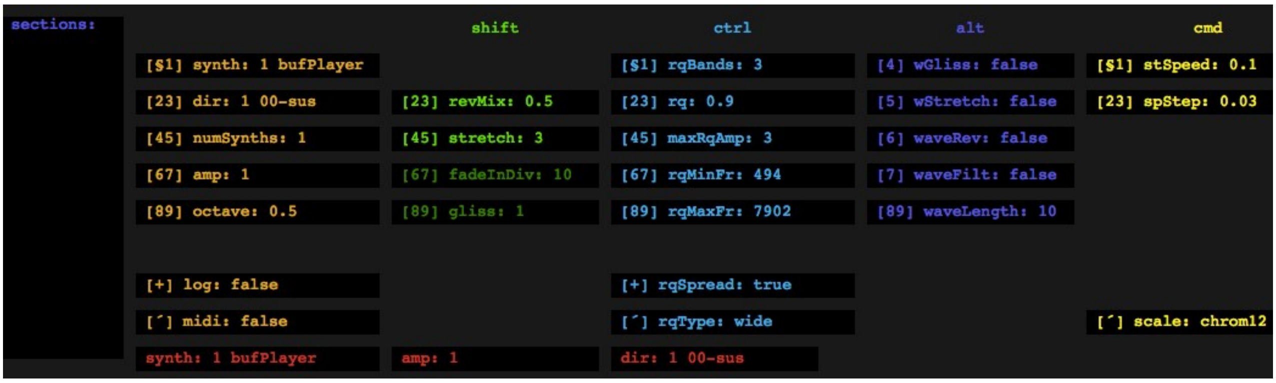
Explanation to the numbers on Figure 17:

- 1: 000-552 Serially filtered and time stretched bus sounds.
- 2: Unmanipulated sounds:
    - 000-180 The MAC’s café on December 9, 2014.
    - 303-450 A totally destroyed sound file. The destruction was accidental but I kept the file and used it in the piece. It is pure distortion.
    - 312-389 The bus recording.
- 3: Laundry rack sounds played with pitches and durations according to the series:
    - 147-244 Time stretched screeching sounds.
    - 195-357 Pinging sounds.

  - ○ 303-388 String sounds with pitch bend.
  - ○ 366-552 Sheet noise sounds.
  - ○ 487-545 Time stretched screeching sounds.
- 4: 270-293 Pinging sound tuned with the instruments tuning.

All sounds were made by playing on my instrument or programmed in SuperCollider. I recorded the output and finished the piece in ProTools.

4.5.5 DESCRIPTION OF SECOND GENERIC INSTRUMENT



sections:		shift	ctrl	alt	cmd
	[$1] synth: 1 bufPlayer		[$1] rqBands: 3	[4] wGliss: false	[$1] stSpeed: 0.1
	[23] dir: 1 00-sus	[23] revMix: 0.5	[23] rq: 0.9	[5] wStretch: false	[23] spStep: 0.03
	[45] numSynths: 1	[45] stretch: 3	[45] maxRqAmp: 3	[6] waveRev: false	
	[67] amp: 1	[67] fadeInDiv: 10	[67] rqMinFr: 494	[7] waveFilt: false	
	[89] octave: 0.5	[89] gliss: 1	[89] rqMaxFr: 7902	[89] waveLength: 10	
	[+] log: false		[+] rqSpread: true		
	[´] midi: false		[´] rqType: wide		[´] scale: chrom12
	synth: 1 bufPlayer	amp: 1	dir: 1 00-sus		

Figure 18: The second generic instrument

This instrument is a bit different from the others. It has the keyboard modifier to use as a label and the colour coding follows the modifier. The button is labelled with the number key to press. You can control which sounds to use, which synth/play method to use and control the chosen synth’s parameters. You can also use an external MIDI controller and use different sounds and play methods on the two keyboards. It's a more advanced version of the first instrument. For a more detailed description of this instrument, see APPENDIX 2.

4.6 *BUS NO. 1* II

(*BUS NO.1* II is not included in this portfolio but can be found in the APPENDIX folder.)

- 2019, recorded instrument version.
- CHANNELS: 8
- DURATION: 9:12
- TOOLS USED: SuperCollider.
- RECORDINGS: The same recordings as in *BUS NO. 1* - I were used, but I added some sounds from High Street in Birmingham recorded on December 9, 2015 with a ZOOM H6.
- STUDIOS USED: EMS in Stockholm.

4.6.1 WORK METHOD

There was only one thing that I wanted to change in *BUS NO. 1* - I, and that was how I used the series. In the first version, I didn't use the original row, the inverted row and the reverse of both as is a common procedure when working serially. The reason for that was that I didn't like the G that appears in the inverted series. (The series comes from my name and there is no G in my name.) When I made this new version of *BUS NO. 1* - I didn't mind the G.

I used the same note series as in *BUS NO. 1* - I but I changed the durations to durations that worked inverted.

Notes:

Original: B E E E A A E B E D E♭ D

Inverted: G D D D A A D G D E E♭ E

Durations:

Original: 29, 9, 9, 9, 21, 21, 9, 29, 9, 12, 15, 12

Inverted: 24.5, 12, 12, 12, 21, 21, 12, 24.5, 12, 9, 15, 9

Here is a plain text illustration of the inversion. Notes to the left, durations to the right:

```
e --------  --------  9
d#------ |  | ------ 15
d ---- | |  | | ---- 12
c#   | | |  | | |
c    | | |  | | |
b -- | | |  | | | -- 29
a# | | | |  | | | |
A  | | | |  | | | |  21
g# | | | |  | | | |
g -- | | |  | | | -- 24.5
f#   | | |  | | |
f    | | |  | | |
e ---- | |  | | ----  9
d#------ |  | ------ 15
d --------  -------- 12
```

4.6.2 DESCRIPTION OF THE PIECE

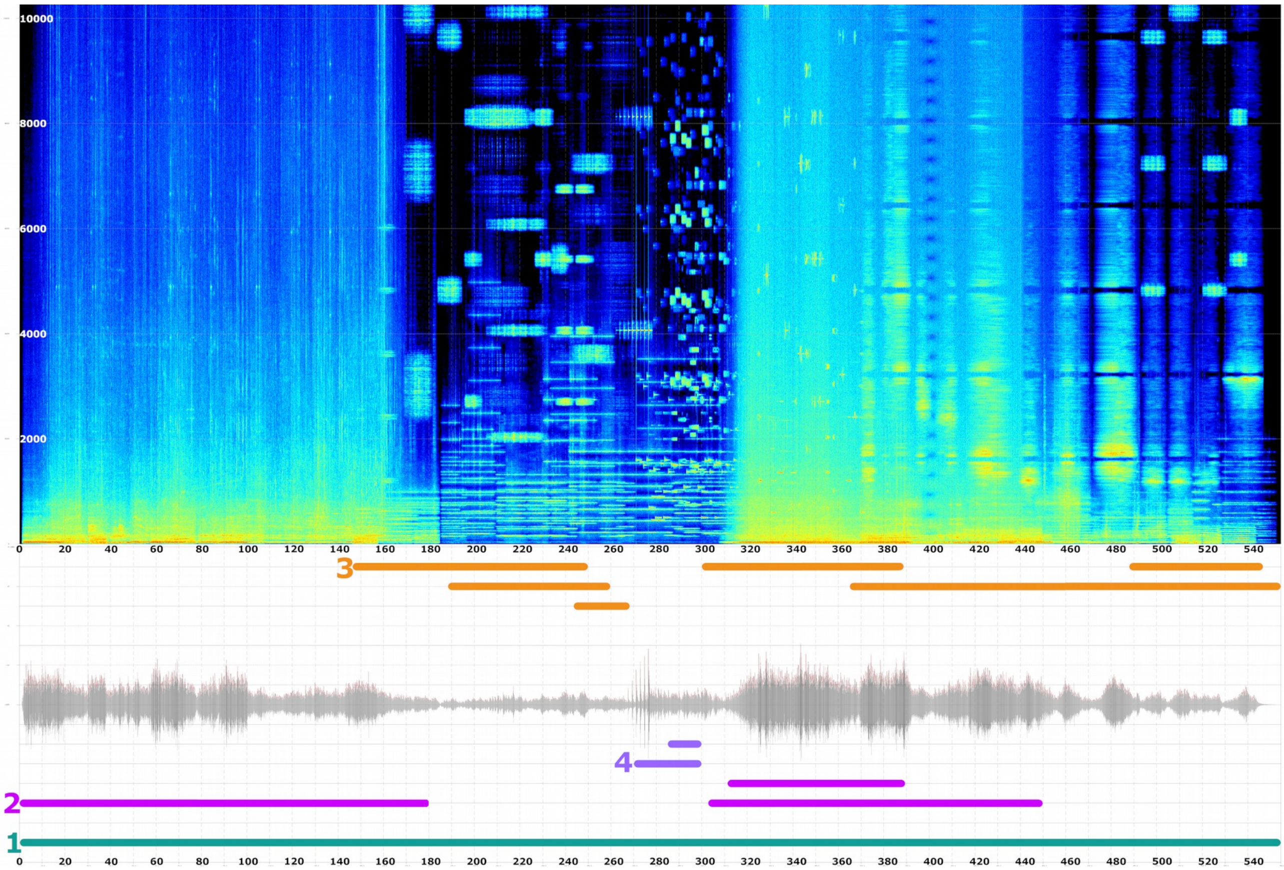


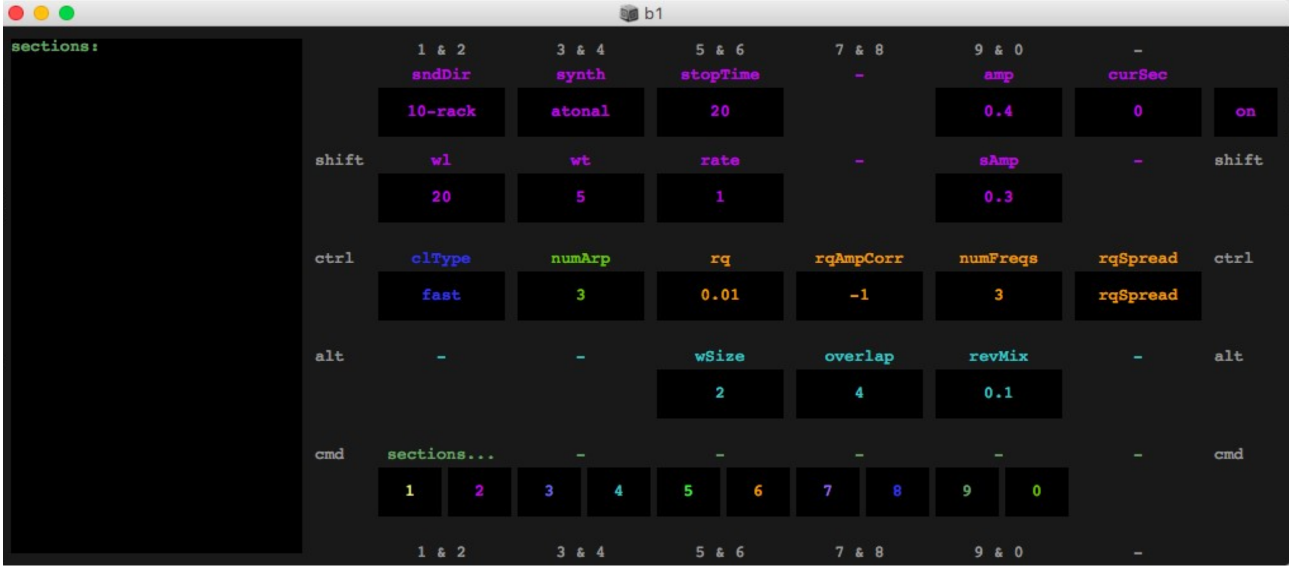
Figure 19: Sonogram of BUS NO. 1 - II

Explanation to the numbers in Figure 19:

- 1: 000-552 Serially filtered and time stretched bus sounds.
- 2: Unmanipulated sounds:
  - 000-180 High Street on December 9, 2015.
  - 303-450 The bus recording.
  - 312-389 The distorted sound.
- 3: Laundry rack sounds played with pitches and durations according to the series:
  - 147-250 Time stretched screeching sounds.
  - 188-260 Time stretched string like sounds.

  - 244-268 Pinging sounds.
  - 300-388 String sounds with pitch bend.
  - 366-552 Sheet noise sounds.
  - 487-545 Time stretched string like sounds.
- 4: Laundry rack sounds:
  - 270-300 Pinging sounds.
  - 285-300 Screeching sounds.

4.6.3 DESCRIPTION OF THE INSTRUMENT



```
b1
sections:
          1 & 2      3 & 4      5 & 6      7 & 8       9 & 0      -
          sndDir     synth      stopTime   -           amp        curSec
          10-rack    atonal     20                     0.4        0          on
shift     wl         wt         rate       -           sAmp       -          shift
          20         5          1                      0.3
ctrl      clType     numArp     rq         rqAmpCorr   numFreqs   rqSpread   ctrl
          fast       3          0.01       -1          3          rqSpread
alt       -          -          wSize      overlap     revMix     -          alt
                                2          4           0.1
cmd       sections...  -        -          -           -          -          cmd
          1    2     3    4     5    6     7    8      9    0
          1 & 2      3 & 4      5 & 6      7 & 8       9 & 0      -
```

Figure 20: BUS NO. 1 - II *instrument GUI*

I had already made an instrument for my piece *AirIac*, which is not included in this portfolio. (The reason for this is that it has no connection to December 9). I thought that this instrument would work well for *BUS NO. 1* - II if I added handling of series. Because of this reuse of an old instrument there are a few things in the instrument that wouldn't have been there if I had written it directly for *BUS NO. 1* - II.

This piece’s identity is the filtered bus sounds, so, when turning the instrument on it starts to play eight serially filtered buses. They can’t be turned off. On top of this you can turn “sections” on and off. A section is made up of a sound directory, a synth and some parameters. Ten sections can be prepared in advance. These sections can be changed. Some parameters like sound type and synth can only be changed when the section is not currently playing. Other parameters like RQ can be changed while the section is playing. The number of different sections is unlimited. The number of currently playing sections is limited to ten. The small buttons with colourful numbers in the bottom of the GUI refer to these (the currently prepared) sections.

I have used colours to see which button belongs to which synth. The synth names have the same colour as their button. For more details about this instrument and instructions on how to play it, see APPENDIX 5.

4.6.4 AFTERTHOUGHTS

When comparing the two versions, again, timing and amplitude control is more exact in the first DAW version. Even though it is almost exactly the same sounds and code in both versions, I don’t think that I have succeeded in making *BUS NO. 1* - II sound as “good” as *BUS NO. 1* - I. The shift from tonal to atonal is not as clear in *BUS NO. 1* - II as in *BUS NO. 1* - I. In ProTools I cut some of my SuperCollider recordings and used some cross-fades. It is of course a question about how much time one spends on the problem.

Did it work? The biggest problem in this piece was that I was determined to use the bus recording, but it sounded awful. I tried many, many different ways of playing it and the serial filter was the best alternative I could find. I could hide the distorting bumps with a narrow RQ. My longing for some tonality was satisfied as the narrow filters make up chords. I also made it possible to retune the laundry rack sounds in the instrument to a tonal scale. So, yes. It worked.

4.7 *INUTI*

- 2016, recorded instrument version.
- CHANNELS: 12
- DURATION: 09:12.
- TOOLS USED: AudioSculpt, SoundStudio and SuperCollider.
- RECORDINGS: A recording of gardeners blowing leaves on William Street in Birmingham on December 9, 2015, and a recorded piano frame, string by string. The piano strings were played with a plectrum.
- STUDIOS USED: EMS in Stockholm and University of Birmingham electroacoustic studios.

4.7.1 CONNECTION TO DECEMBER 9

In *INUTI* the connections are the recording of the gardeners made on December 9 and the piece length: 9:12.

4.7.2 IDEA BEHIND THE PIECE

After having so much trouble with the aurally uninteresting bus recording in my previous piece I was not very enthusiastic about the leaf blowing gardeners I had for this piece. What should I do with it? I thought about *UNDER THE EIFFEL TOWER* - I and wondered if I could let the recorded piano play the sound spectrum of the recorded gardeners. While I experimented with this in different ways I ended up using – not the sound of the gardeners, but a partial analysis (made in AudioSculpt) of the gardeners' sounds. I made functionality to play the data from the analysis in different ways and I controlled the mechanism with global variables. I had trouble remembering which variables were set to true and which to false. To solve this I made a GUI that showed me what was on and what was off. This was how I got the idea to make instruments of my pieces.

Like in *Bus No 1* I wanted to experiment with tonality so the piece can be tonal, atonal or a mix of both.

4.7.3 WORK METHOD

In *INUTI* I used selections of all partials to use in separate parts of the piece. These are the names I gave the selections:

- *Piano* – The shortest partials. Maximum duration was two, which means only two partials in a row from the SDIF file. 2 * 0.005805 = 0.01161 seconds long. The selection has 4677 partials.
- *Low* – The lowest frequencies from the SDIF file. I chose all frequencies below 43 Hz. The selection has 309 partials.
- *Env* – The longest partials. I chose partials with minimum length 90 (breakpoints), which means that they have to occur 90 times in a row, which means approximately 90 * 0.005805 = 0.52245 seconds long. In the Env-part of *INUTI* I use all the information from the partial analysis to make envelopes according to the SDIF data. This makes the tones unsteady in pitch. I prolong the envelopes by 0.5 second to make them fade out more softly. The selection has 138 partials.
- *Long* – This selection was made from a time grid. Simply one partial each fifth second. The selection has 109 partials.
- *Trill* – This one was also made from a time grid. One partial each ninth second. The selection has 61 partials.

4.7.4 DESCRIPTION OF THE PIECE

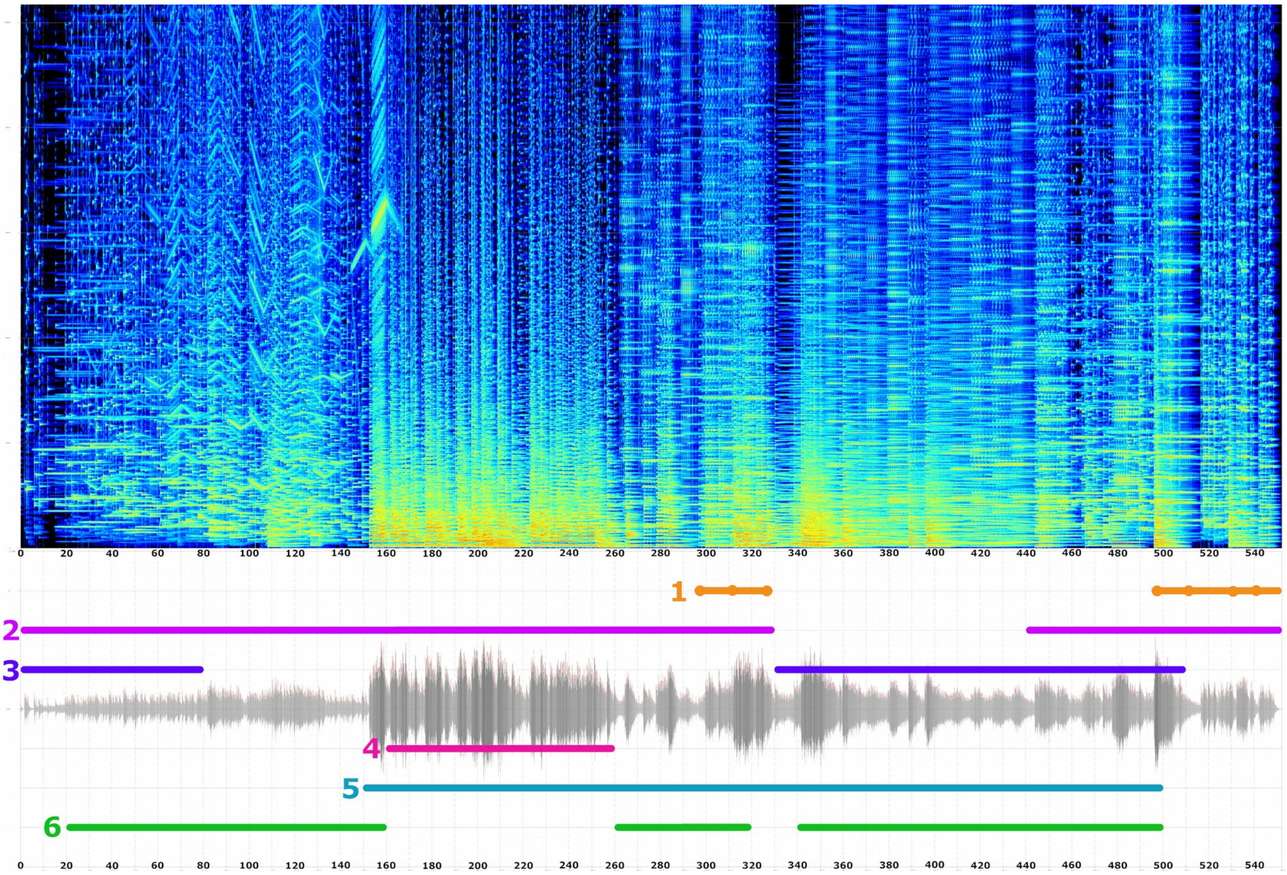


Figure 21: Sonogram of INUTI

Explanation to the numbers in figure 21:

1 Tonality:

- 000 – 294: No tonality. Play the SDIF information unaltered.
- 295 – 329: Transpose frequencies to the nearest frequency from a Huzam scale.
  - 295 – 309: Use all notes from the scale (in any octave).
  - 310 – 324: Use only notes B, D and F.
  - 325 – 329: Use notes A, C and E.
- 330 – 494: No tonality.
- 495 – 539: Transpose frequencies to the nearest frequency from a Huzam scale.

    - 495 – 510: Huzam: A, C and E.
    - 510 – 529: Huzam: Use all notes from the scale (in any octave).
    - 530 – 539: Huzam: A, C and E.
- 540 – 552: Transpose frequencies to the nearest frequency from a C major scale. Use notes C, E and G.

2 Piano-selection:

- 000 – 059 Piano with modulus 5 and HPF 300.
- 060 – 079 HPF 200.
- 080 – 099 HPF 100.
- 100 – 149 HPF 0.
- 150 – 279 modulus 10.
- 280 – 329 modulus 7.
- 440 – 459 modulus 4.
- 460 - 499 modulus 3.
- 500 – 552 modulus 1.

3 Long-selection

- 000 – 080 amplitude 0.3.
- 330 – 359 amplitude 0.5, LPF 246.952.
- 360 – 369 amplitude 0.4.
- 370 – 439 amplitude 0.3.
- 440 – 510 amplitude 0.4, no LPF.

4 Env-selection:

- 160 – 260 amplitude 1, modulus 1.

5 Low-selection:

- 150 – 500 amplitude 1, modulus 1.

6 Trill-selection:

The Trill-selection can be played with two different synths /play methods called Trill and Arp.

- 020 – 160 Trill with amplitude starting at 0.1 and ending at 0.7. Modulus 1.
- 260-320 Arp with amplitude starting at 0.7 and ending at 1. Modulus 1.
- 340 – 500 Arp at amplitude 1 and modulus changing from 1 to 2.

4.7.5 DESCRIPTION OF THE INSTRUMENT

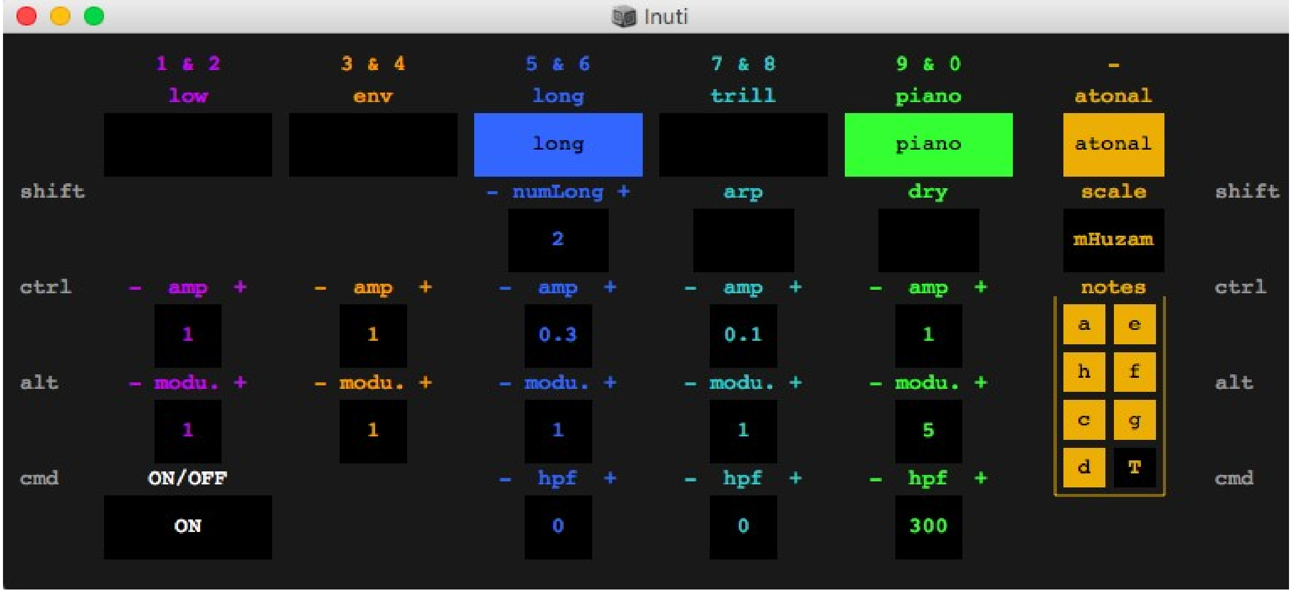


Figure 22: INUTI instrument GUI

In the instrument the modu. (modulus) is used to control the density of the played tones. A higher modulus value means a less dense tone mass, a lower modulus means more dense. Modulus 1 means to play everything from the selection.

The instrument has a button called HPF (high pass filter). When it has a value (other than zero) it means that the instrument will transpose all SDIF frequencies so many octaves up as is required to make the SDIF frequency higher than the HPF value. The instrument will not play frequencies below the HPF frequency.

On the top row below the key labels are the selections described above in WORK METHOD.

All selections or play types (low, env, long, trill and piano) have an amplitude and a modulus button. Long, Trill and Piano have an HPF button. Long has a numLong button where the number of simultaneous long tones can be controlled. Trill has an Arp button which changes the Trill into an arpeggio. Piano has a dry button which makes the tones short. I haven't used the dry button in the submitted piece.

The orange column on the right controls the tonality. The atonal button is a toggle button. When atonal is off the instrument uses the scale selected with the scale button below. Individual notes from the chosen scale can be selected by using the control key and the corresponding character on the keyboard.

See APPENDIX 6 for a more detailed description of the instrument and how to play it.

4.7.6 AFTERTHOUGHTS

Making this first piece specific instrument led me to make the rest of the piece specific instruments in this portfolio. I had for a long time wanted to stop using DAWs and like this I could.

4.8 *INANNAN*

- 2019, recorded instrument version.
- CHANNELS: 12
- DURATION: 12:09.
- TOOLS USED: AudioSculpt, SuperCollider.
- RECORDINGS:
  - A recording from 1995 where Amit Sen plays cello.
  - A recording of a walk along the Centralbron in Stockholm, Sweden on December 9, 2016. This recording was made with a ZOOM H6.
- STUDIOS USED: EMS in Stockholm.

4.8.1 CONNECTION TO DECEMBER 9

The connection is the Centralbron recording made on December 9, 2016 and the piece length: 12:09.

4.8.2 IDEA BEHIND THE PIECE

In 2017, after struggling with another (still unfinished) piece (called *FIVE SPECIES*) I felt that I had to do something else for a while. I looked through my DAT tapes and found an old piece of mine called *Inanna*.

Inanna is for cello and tape. In 1995 I contacted cellist Amit Sen who came to EMS and I recorded him playing both a notated cello part and a lot of other cello sounds that I wanted to use in a tape part. “Annan” means “other” in Swedish. The name *INANNAN* is “another *Inanna*”.

I had previously experimented with “slicing” (see 4.8.3 below) sound files in order to be able to retune them. I tried this technique on a part of the recorded solo cello melody and thought that this might be something interesting to explore. For me it would be the first time in many years that I had worked with a melody. The INANNA melody is my own creation. I would probably find it hard to work with someone else's melody (independence again), except – as I have done later – if the composer is unknown[12].

4.8.3 WORK METHOD

INANNAN is an instrument and the piece in this portfolio is a recording of an automated instrument version.

In AudioSculpt I "sliced" the melody recording. I made twelve sound files, one for each of the twelve tones of a western 12-ET scale. I used my eyes to do this. I looked at, for instance a C and made a filter for the C and all it's visible overtones. Then a C# and so on. This was time consuming and I have later found a way to do it more automatically, but this "hand made" slicing sounds different. I had intended to have a piece that used the auto-slicing in this portfolio, but changed my mind.

INANNAN is more discovered than composed. When the slicing was done I played the slices in a similar way like I did in *INUTI*. I used a fundamental analysis of *Inanna's* melody and played a random number of slices simultaneously. When I heard it the first time I was very surprised. I know the melody Amit plays very well (I composed it) but it is unrecognisable when it's played back like this. It is also very difficult to memorize. It sounds quite new every time I hear it. I had no idea that it would sound as it does before I heard it. That's why I consider it discovered rather than composed.

On top of this sliced melody I use the other cello sounds that I recorded with Amit. They are grouped in four types: fast tremolo (fTrem), slow tremolo (sTrem), single tones (tone) and pizzicato (pizz).

[12] I use the hymn *O Come, O Come, Emmanuel* in a piece called *Lemuriel* in which I make recorded Lemurs sing the hymn.

These cello sounds start automatically depending on a few variables. They can also be started or turned off by the instrument user.

4.8.4 DESCRIPTION OF THE PIECE

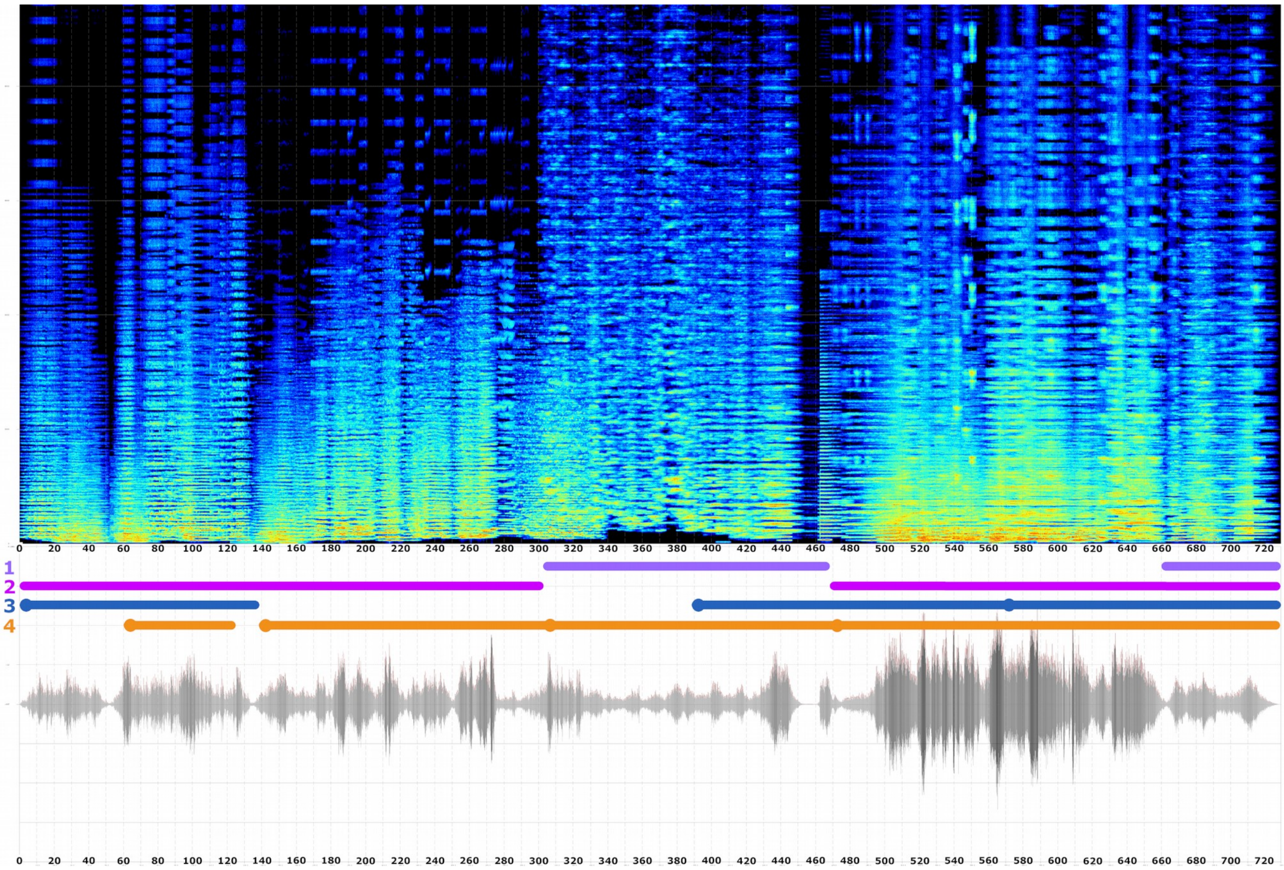


Figure 23: Sonogram of INANNAN

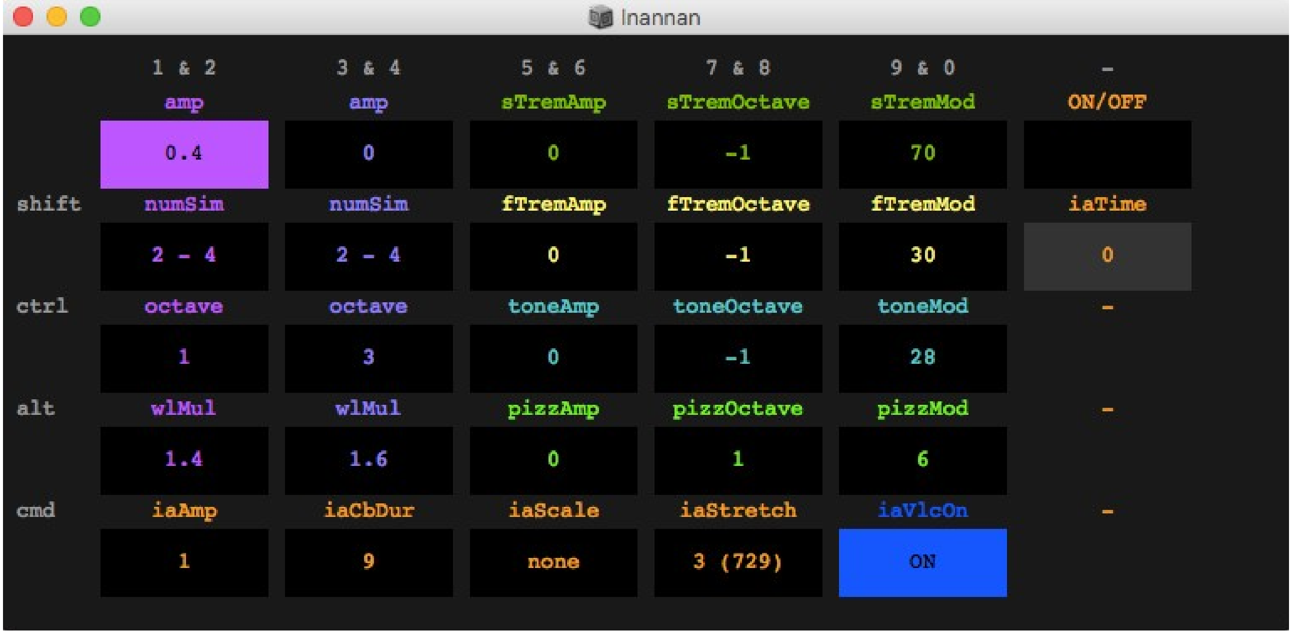
Description of the numbers in Figure 23:

- 1 & 2:
  - INANNAN uses two play methods (i.e. SuperCollider synths) for playing the sliced files, one playBuf (which plays the sounds as they are) and one time stretch. Number 1 in the image above is the Stretch and number 2 is the playBuf. The playBuf plays a portion of the sliced files. The length is decided by the SDIF file. The length gets multiplied by the wlMul value to avoid silences between playback. The number of simultaneous sliced files is random. Playback always starts at startTime / 3 (iaStretch) in

the file. The stretch synth uses time stretch so the sounds are not repeated as they are in the playBuf.

- 3:
    - Number three shows which scale is used. It starts with susato[13] followed by no scale, A# minor and ends with 12-ET.
- 4:
    - Number four is the tape part cello recordings. First pizzicatos, followed by fast tremolos, plain tones and ending with slow tremolos.

4.8.5 DESCRIPTION OF THE INSTRUMENT



	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-
	amp	amp	sTremAmp	sTremOctave	sTremMod	ON/OFF
	0.4	0	0	-1	70	
shift	numSim	numSim	fTremAmp	fTremOctave	fTremMod	iaTime
	2 - 4	2 - 4	0	-1	30	0
ctrl	octave	octave	toneAmp	toneOctave	toneMod	-
	1	3	0	-1	28	
alt	wlMul	wlMul	pizzAmp	pizzOctave	pizzMod	-
	1.4	1.6	0	1	6	
cmd	iaAmp	iaCbDur	iaScale	iaStretch	iaVlcOn	-
	1	9	none	3 (729)	ON	

Figure 24: INANNAN instrument GUI

The two columns to the left (cerise and lilac) control the playing of the melody slices. They are read from top to bottom. The rest of the GUI is read from left to right. The four top rows control the cello sounds, sTrem, fTrem, tone and pizz. The bottom row control more global parameters such as *INANNAN's* amplitude and stretch length.

[13] Susato is a scale made by myself. It is derived from Tielman Susatos piece Ronde och Saltarello

In *INANNAN,* an AudioSculpt fundamental frequency analysis of the original *Inanna* melody is used. The melody is 4:03 long. I exported it to an SDIF file and use the SDIF data in my instrument. When the *INANNAN* instrument is started, it starts to loop over the SDIF data. It reads startTime, frequency and duration. The duration gets stretched to variable iaStretch, which with the default value 3 becomes 12:09.

During the loop the instrument checks whether a scale and/or an octave is chosen. If it is, it moves the SDIF frequency to the desired value. If nothing is chosen it plays the frequency from the SDIF file.

The startTime value decides from where in the sound file the instrument starts playback, so during the piece the instrument will step through the sliced sound files from the beginning to the end.

When the instrument is started the sliced sound files is read into an array which also has information about the files' fundamental frequency. In the loop the instrument chooses a random sound file and transposes it to the desired frequency like this: playBackRate = desiredFrequency / theSoundFilesFundamentalFrequency. In the GUI the user can choose numSim which is a range, for example 2 – 4. The actual numSim is randomly chosen from this range.

The loop plays the playBuf synth, the stretchSynth, both or none. It also checks whether iaVlcOn is true and if the SDIF duration value if bigger than iaCbDur. If it is, it starts the function iaVlc which plays the cello sounds. These variables can be changed by the user.

The vlc loop uses time to decide which cello sound to play like this:

- if time < 138: play the pizzicato (pizz) sounds
- if time < 303: play the fast tremolos (fTrem)
- if time < 468: play the cello tones (tone)
- if time >= 468: play the slow tremolos (sTrem)

The vlc loop works like the main loop but it uses another SDIF file. It is an AudioSculpt partial analysis SDIF file of the above mentioned Centralbron walk. NumSim, scale and octave works like in the main loop.

See APPENDIX 6 if you want to know how to play the instrument.

4.8.6 AFTERTHOUGHTS

The original plan to use a recorded choir and five species counterpoint has until now not been very successful, as it is still not finished. But, I got quite close to the dreamt skewing string orchestra (mentioned in the INTRODUCTION). The way it sounds made me very enthusiastic and I will experiment more with this in the future. I have used the same procedure in another piece and it doesn't sound interesting at all so the original source file is important. It can't be any sound. In my coming experiments I will try to get more precise control over the exact frequencies I produce over time and how they fit together.

4.9 *MIŠKAS*

- 2019, recorded instrument automated version.
- DURATION: 12:09.
- CHANNELS: 12
- TOOLS USED: SuperCollider.
- RECORDINGS:
    - Recordings of the forests surrounding Druskininkai in Lithuania made with a ZOOM video recorder and a Marantz.
    - Recordings from a concert hall made on December 9, 2017 with an iPhone 8.
- STUDIOS USED: EMS in Stockholm and University of Birmingham electroacoustic studios.

4.9.1 CONNECTION TO DECEMBER 9

In this piece the connections are the recording made on December 9, 2017 and the piece length, 12:09. In the summer of 2012 I spent a month in the small Lithuanian town Druskininkai. I had been invited for the DAR residency[14]. Half of the sounds available in the *MIŠKAS* instrument are recordings from the forests surrounding Druskininkai. The forests were very beautiful and I intended to also make a video – therefore the video recorder, but the video still awaits its completion. Miškas means forest in Lithuanian. The other half are recordings from a concert space made on December 9, 2017.

4.9.2 IDEA BEHIND THE PIECE

MIŠKAS has a long history. It has evolved since 2012. The original plan was simply to make a piece that used sounds from Druskininkai (as was required of me for the residency.). I experimented with

[14] See: https://www.artdruskininkai.lt/en/about/. August 10 2019.

the *Stria* filters I mentioned in the INTRODUCTION and found out that they sounded really nice when used on my forest recordings. I did what I used to do in those days. Recorded my SuperCollider output and put the piece together in ProTools.

In 2013 I was asked to make a longer version of *MIŠKAS* for a sound installation in Mexico City. For this I rewrote the code and made a new extra long version.

In 2014 I was asked by dancer and choreographer Marie Louise Crawley to make a new *MIŠKAS* like piece to be played live in a dance performance. For this I rewrote the code again and made a version that could be played live by running code. The name of the performance became *The Forest.* It was performed with dancers Marie Louise Crawley and Madalena Brak-Lamy at the MAC (Midlands Art Center, Birmingham, UK) on December 12 2014. This version of *MIŠKAS* had new forest recordings and lots of other sounds. For example sounds of playing children, rain being turned in to a string instrument and birds.

In 2017 I used my *MIŠKAS* code and sounds from VISEO RURAL[15]s sound archive and made a piece called *Pastoral.* VISEO RURAL held a composition composition. The requirement was to use their binaural recordings from Viseo in Portugal. This is the only time I have ever sent something to a competition. *Pastoral* got second prize.

In 2018 I was asked to make a piece with sounds from aircrafts with recordings from the sound archive Make history heard[16]. This time I used the by now well used code to make an instrument with a GUI. The piece called *AirIac* was performed live on Linköpings flygmuseum in Sweden.

I made *AirIac* and its instrument during a residency in May 2018 at Inter Arts Center[17] (IAC) in Malmö.

[15] https://www.viseururalmedia.org/
[16] http://www.soundsofchanges.eu/
[17] https://www.iac.lu.se/

In 2019 I made a new simpler instrument dedicated to *MIŠKAS.* In the submitted version I use the original Druskininkai sounds and, of course, a recording from December 9, 2017.

4.9.3 WORK METHOD

Technically *MIŠKAS* is a very simple piece. It can, and has been used as a sound installation.

In the *MIŠKAS* instrument, the sounds are played back either as they are or filtered according to a randomly chosen scale. The RQ value wanders in a direction chosen by the player. That is all. The scales are:

- Common 12-ET. Start frequency: 110, octave ratio: 2, steps per octave: 12.
- Golden mean scale (similar to the one John Chowning used in *Stria*[18]). Start frequency: 136, octave ratio: 1.618, steps per octave: 9.
- Odd scale invented by myself. Start frequency: 120, octave ratio: 2, steps per octave: 53. All 53 steps are not used, only: 5, 12, 21, 30, 34, 43, 52.

[18] See for example: Matteo Meneghini: *STRIA*, BY JOHN CHOWNING: ANALYSIS OF THE COMPOSITIONAL PROCESS: https://pdfs.semanticscholar.org/0eee/b69708fb5510e1307e133ea3f6db1500c632.pdf. August 10 2019.

4.9.4 DESCRIPTION OF THE PIECE

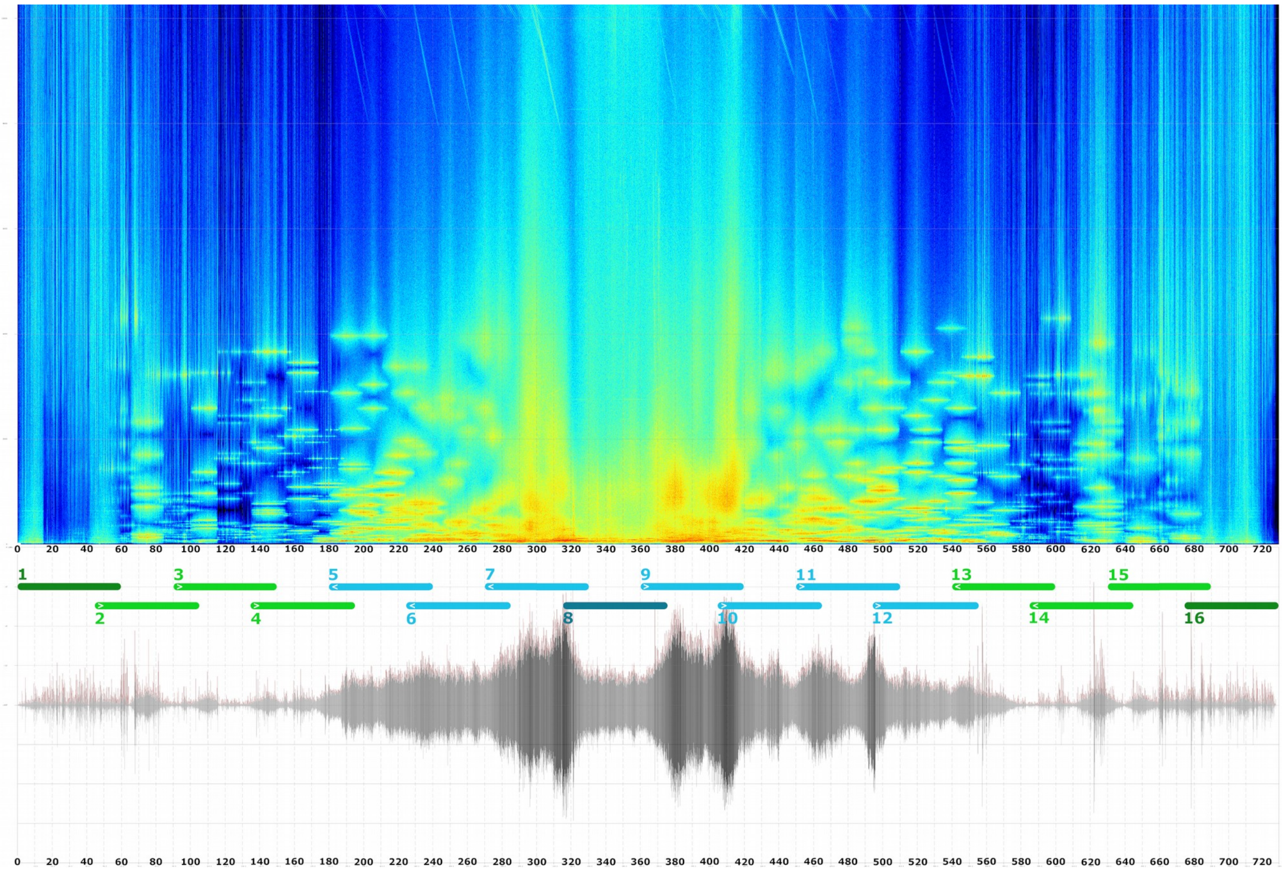


Figure 25: Sonogram of MIŠKAS

Explanation to Figure 25:

- The sign > means that the RQ value gets smaller and smaller, making the sounds thinner. < is the opposite. (The signs can be seen in the numbered coloured lines in the image above. They are white and very small. They can also be seen in the numbered list below.)
- Green means crackling sounds, turquoise means forest sounds.
- Dark colour means unfiltered sounds.

Explanation to the numbers in Figure 25:

1 Unfiltered crackling sound.

2 Filtered crackling sound. RQ > from 0.055 to 0.015

3 Filtered crackling sound. RQ > from 0.0085 to 0.0045

4 Filtered crackling sound. RQ > from 0.0025 to 0.00075

5 Filtered forest sound. RQ < from 0.0085 to 0.035

6 Filtered forest sound. RQ < from 0.055 to 0.095

7 Filtered forest sound. RQ < from 0.025 to 0.65

8 Unfiltered forest sound.

9 Filtered forest sound. RQ > from 0.9 to 0.5

10 Filtered forest sound. RQ > from 0.3 to 0.08

11 Filtered forest sound. RQ > from 0.06 to 0.02

12 Filtered forest sound. RQ > from 0.009 to 0.005

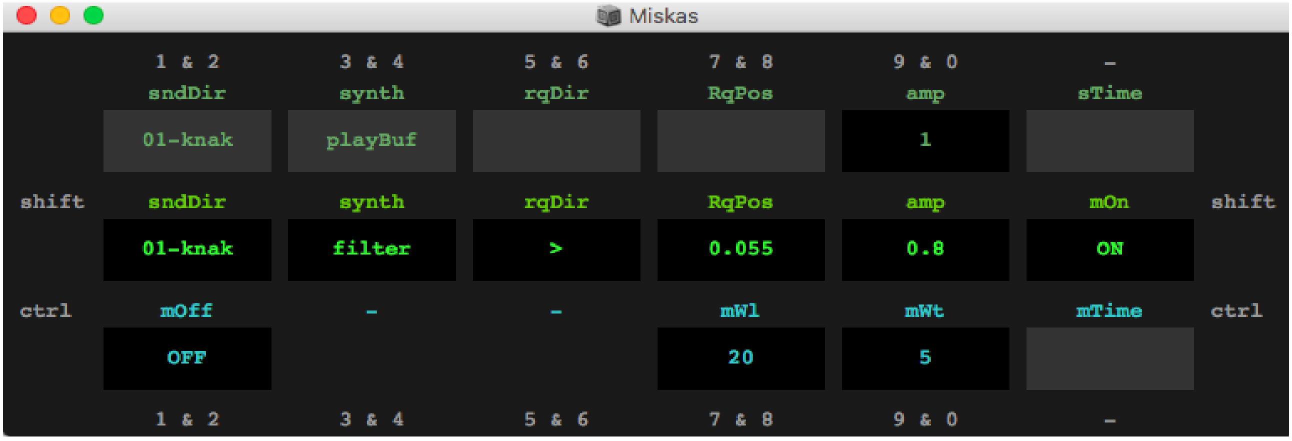
13 Filtered forest sound. RQ > from 0.06 to 0.1

14 Filtered crackling sound. RQ < from 0.3 to 0.7

15 Filtered crackling sound. RQ steady at 0.055

16 Unfiltered crackling sound.

4.9.5 DESCRIPTION OF THE INSTRUMENT



	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	
	sndDir	synth	rqDir	RqPos	amp	sTime	
	01-knak	playBuf			1		
shift	sndDir	synth	rqDir	RqPos	amp	mOn	shift
	01-knak	filter	>	0.055	0.8	ON	
ctrl	mOff	-	-	mWl	mWt	mTime	ctrl
	OFF			20	5		
	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	

Figure 26: MIŠKAS instrument GUI

In *MIŠKAS* you run “sections”. A section is a sound directory, a play method, an amplitude, if the synth is a filter it is also an RQ direction and an RQ start value. You can only run one section at a time. In the top row in the GUI you see what is currently running (or will run if you haven’t started *MIŠKAS* yet). The second row shows the values for the next section. These values can be edited until you start this next section (by pressing shift + the minus sign). The bottom row can only be changed while *MIŠKAS* is off. These buttons control *MIŠKAS* on/off, mWl, mWt (wave length, wait time – the same as in previous instruments) and mTime. sTime shows how long current section has been running, mTime shows how long *MIŠKAS* has been running.

For a full explanation of how to use the instrument, see APPENDIX 8.

4.9.6 AFTERTHOUGHTS

The instrument is easy to play. In my opinion the sounds and play methods sit well together. I find the piece relaxing both to hear and play.

5. CONCLUSION

My compositional development during these years went from using my first generic instruments, recording the output and putting the piece together in a DAW. I spent many hours editing my sounds in AudioSculpt before using them. In my third piece *BUS NO. 1* I strived for and achieved control over tonality/atonality. A bigger change happened in the fourth piece *INUTI* when I realised that I could use the SDIF file as a score. When playing the SDIF I suddenly had very, very many sounds and my job became to filter the output. I didn't have to create the output - I had too much and had to remove parts of it. In the fifth piece *INANNAN* I found the sound file slicing very interesting. I will continue to work with SDIF files and slicing of sounds. I have also tried to work directly with sonograms (insted of partial analysis SDIFs) but that hasn't been successfull so far.

I have achieved to leave DAWs behind me and I've come quite near to the dreamt sound of the skewing string orchestra. Some experiments have been successful, others not. I have learned a lot and have many more tools in my toolbox for use in the future. It has been six years well spent.

In *NINE STATIONS* I I tried to make scales based on field recordings. I imagined that I would be able to find each station’s nine strongest frequencies and that the gradual filters would make the nine frequency chords emerge and disappear. I think this was successful. The other idea to tune the characteristic sounds to the same nine frequency scale was not so successful. Some sounds sounded “bad” when transposed. In many of the characteristic sounds it was impossible to find a frequency to use as a fundamental. With these sounds I chose for example 100 Hertz and transposed accordingly. But however I did the transposition, the sounds seldom sounded connected to the station’s ambient sound.

NINE STATIONS II – This was the second dedicated instrument I did. I made the instrument so that it could recreate and improve the original piece. It was an interesting project but the instrument is difficult to play. At least if you are in a concert situation with a time limit and a plan for what you want to achieve. It is not difficult to just “play” with the instrument. I learnt a lot by making this instrument and I think that version two of *NINE STATIONS* is better then the first one.

In *UNDER THE EIFFEL TOWER* I wanted to examine spectrum manipulations, to turn sounds inside out and upside down. I also wanted to use my field recording made under the Eiffel Tower. I made the piece according to plan in an unusually short time. I consider the project successful.

UNDER THE EIFFEL TOWER II - I made this instrument after the *NINE STATIONS* - II instrument. This instrument was also made after the piece and was designed to be able to recreate the piece. It can and in this respect it was successful. I don’t think the new SDIF part is very successful. The second version of the piece still needs more work.

BUS NO. 1 I - I longed for tonality and this longing was satisfied. I added the possibility to play tonal on my second generic instrument and I like the switch from tonal to atonal that comes in the middle of the piece.

INUTI was a very interesting project. I had a wish to be surprised by my instruments and this time I was. It was very interesting to work with the SDIF data, to make different selections of partials and have them played in different ways. I didn’t have to decide what and when to play something. I only had to choose what of all the data to play. This was the first piece dedicated instrument. In the previous pieces I had produced all sounds with code in SuperCollider but to finish the pieces I recorded my SuperCollider output and put it all together in a DAW. In *INUTI* I had no need for a DAW.

INANNAN - My intended piece *FIVE SPECIES* is still not finished but all the work I put into it was used in other pieces. The slicing of sound files for example. I sliced the recording of the choir singers and I examined many, many ways to try to be inside sounds by playing its SDIF data in many different ways. In *INANNAN* you are technically inside the sound as the sound is sliced and played in different speakers. I was very surprised when I listened to the result for the first time thinking: What is this? I listened over and over again but I could not really understand what I heard, but I liked it.

MIŠKAS - In 2012 I loaded the Druskininkai forest recordings into old filtering code that I had and discovered that it sounded nice. The piece has evolved many times since then. In the 2019 version that is included in this portfolio I added recordings from a concert hall and made the instrument, which became the easiest and nicest to play.

5.1 FUTURE DIRECTIONS

I will return to Linux and make my second generic instrument work in my favourite OS again. I will try to enable it to play all my pieces, my old pieces as well as new. I also want to have all my sounds organised in one directory and write functionality to prevent me from using wrong sound type for chosen play method. I will also need functionality to change loaded sounds as I probably won’t be able to have all loaded simultaneously. I will probably continue exploring different ways of “slicing” sound files and explore more ways to use SDIF files.

I would like to be able to play long (about 60 minutes long) “meditations” or “sound journeys” in which I could play parts of all my pieces – not as they are now, but to combine them quite freely. To be able to do this I need a generic instrument that can produce various “sound worlds”.

BIBLIOGRAPHY

Alpern, A., 1995. Techniques For Algorithmic Composition Of Music.[online] Citeseerx.ist.psu.edu. Available at: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.23.9364&rep=rep1&type=pdf> [Accessed 7 October 2020].

Barlow, C., 2011. Algorithmic Composition, Illustrated By My Own Work: A Review Of The Period 1971-2008. [online] Clarlow.org. Available at: <http://clarlow.org/wp-content/uploads/2014/03/Algorithmic-Composition.pdf> [Accessed 7 October 2020].

Baudouin, O. 2007. A reconstruction of Stria. [online] www.mitpressjournals.org: Available at: <https://www.mitpressjournals.org/doi/abs/10.1162/comj.2007.31.3.75> [Accessed 7 October 2020].

Blackwell, A., and Collins, N., 2005. The programming language as a musical instrument. In: P. Romero, J. Good, A. Chaparro and S. Bryant, eds., Proceedings of Psychology of Programming Interest Group. [online] PPIG2005. pp.120–130. Available at <https://ppig.org/files/2005-PPIG-17th-blackwell.pdf> [Accessed 7 October 2020].

Bell, R., 2013. An Approach To Live Algorithmic Composition Using Conductive. [online] Lac.linuxaudio.org. Available at: <http://lac.linuxaudio.org/2013/papers/49.pdf> [Accessed 7 October 2020].

Chadabe, J, 1984. Interactive Composing: An Overview. Computer Music Journal, vol. 8, no. 1, 1984, pp. 22–27. Available at: <JSTOR, www.jstor.org/stable/3679894>. Accessed 7 Oct. 2020.

Couturier, J., n.d. ETHERACTION: PLAYING A MUSICAL PIECE USING GRAPHICAL INTERFACES. [online] Recherche.ircam.fr. Available at: <http://recherche.ircam.fr/equipes/repmus/SMC04/scm04actes/P34.pdf> [Accessed 7 October 2020].

Edwards, M., 2012. slippery chicken, a specialised algorithmic composition program. [online] Michael-edwards.org. Available at: <https://michael-edwards.org/sc/media/sc-paper-long.pdf> [Accessed 7 October 2020].

Lewis, G E, 2000. Too Many Notes : Complexity And Culture In Voyager. [online] Pdfs.semanticscholar.org. Available at: <https://pdfs.semanticscholar.org/f212/01a99a3b5a0bb6146e2ebbb0a18c9e527208.pdf> [Accessed 7 October 2020].

Lopez-Lezcano, F., 2010. CATMASTER AND "A VERY FRACTAL CAT", APIECE AND ITS SOFTWARE. [online] Ccrma.stanford.edu. Available at: <https://ccrma.stanford.edu/~nando/publications/catmaster_smc2010.pdf> [Accessed 7 October 2020].

Lopez-Lezcano, F., 2013. CEC — Econtact! 13.2 — A Very Fractal Cat By Fernando Lopez-Lezcano. [online] CEC | Canadian Electroacoustic Community. Available at: <https://econtact.ca/13_2/lopez-lezcano_fractal.html> [Accessed 7 October 2020].

Thigpen, B., 2020. CEC — Econtact! 11.4 — Spatialization Without Panning - Benjamin Thigpen. [online] CEC | Canadian Electroacoustic Community. Available at: <https://econtact.ca/11_4/thigpen_spatialization.html> [Accessed 7 October 2020].

Truax, B., 1988. Real-Time Granular Synthesis with a Digital Signal Processor. Computer Music Journal, 12(2), p.14. Available at: <https://www.jstor.org/stable/3679938>, [Accessed 7 October 2020].

Ungvary, T:

- (1988). REPORTS K.A.C.O.R. Kineto-Audiologic Communication Research Group. Tamas Ungvary and Simon Waters. Visual documentation of structural information: a tool for composition and analysis in electro-acoustic music. 1/88.
- (1989). REPORTS K.A.C.O.R. Kineto-Audiologic Communication Research Group. Tamas Ungvary and Simon Waters. The SONOGRAM : A tool for composition and analysis in electro-acoustic music. 4/89.
- (1989). REPORTS K.A.C.O.R. Kineto-Audiologic Communication Research Group. Peter Lunden, Tamas Ungvary. SONOGRAM 7.0 technical description. 14/89.
- (n.d.). CHOR3E

Zmölnig, Io., and Eckel, G., 2007. LIVE CODING: AN OVERVIEW. In: ICMC Proceedings. [online] International Computer Music Conference. Copenhagen, Denmark, pp.295–298. Available at: <http://quod.lib.umich.edu/i/icmc/bbp2372.2007.063/ 1> [Accessed 7 October 2020].

APPENDICES

APPENDIX 1

HOW TO PLAY THE INSTRUMENTS

These instruments works on Mac OSX only.

This text is common for all instruments in this portfolio. The letters *XX* should be exchanged for the piece/instruments name or its abbreviation.

- Before you start, you have to have SuperCollider installed and the directory HhClasses in SuperColliders Extension folder. On OSX this is usually in /home/*username*/Library/Application Support/SuperCollider/Extensions/.
- Unpack HH-*XX*.tgz somewhere.
- Start the SC-IDE.
- All pieces consists of a number of numbered scd-files and the directory sounds where all source sounds are located. There might also be a folder with SDIF files.
- From the SC-IDE, open all scd-files.

01-*XX*Start.scd

- Run the first START code block to load the sounds and start the server.
  - The post window should report which sounds are loaded and how many.
- Skip the STOP block. Run this when you are finished and want to quit.
- Run the SYNTHS block
- Scroll down and choose which OUT block you want to run. The choices are usually:
  - 12 (low ring of 8 speakers, high ring of 4 speakers)
  - 8 (ring of 8 speakers)
  - 6 (ring of 6 speakers)
  - 2 (stereo).
  - All rings start with 0 as front left and continues clockwise in a circle. 0 followed by 1 to the right etc.

02-*XX*Data.scd

Run the code. You should see this: *XX* data ok

03-*XX*.scd

Run the code. You should see: *XX* functions ok

04-XXGui.scd

Run the code. You should see the instrument GUI.

05-*XX*Auto.scd

This file is optional. It is an automated version of the piece. If you want to run it, just run the code block and then the line ~*XX*Auto.start. When it's running it will show you what is happening in the GUI.

After this, read the appendix you are interested in:

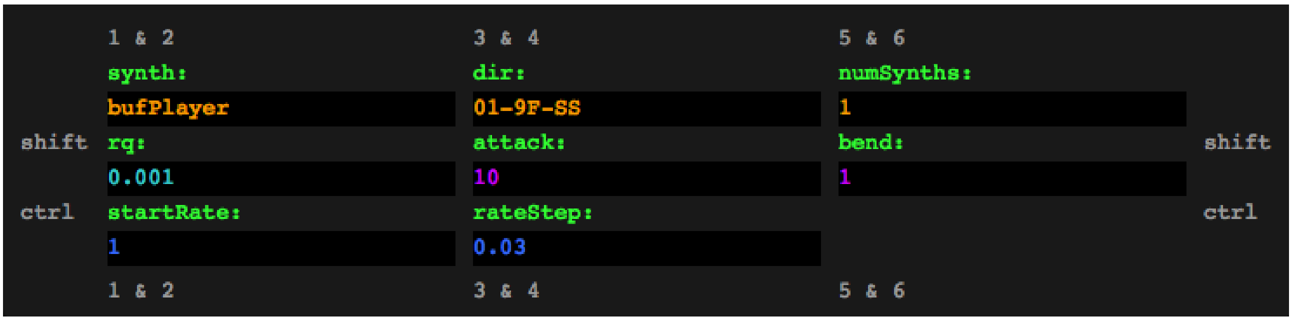
- Appendix 2 – The two generic instruments.
- Appendix 3 – *NINE STATIONS* - II – THE INSTRUMENT AND HOW TO PLAY IT.
- Appendix 4 – *UNDER THE EIFFEL TOWER* - II – THE INSTRUMENT AND HOW TO PLAY IT.
- Appendix 5 – *BUS NO. 1* - II – THE INSTRUMENT AND HOW TO PLAY IT.
- Appendix 6 - *INUTI* – THE INSTRUMENT AND HOW TO PLAY IT.
- Appendix 7 - *INANNAN* – THE INSTRUMENT AND HOW TO PLAY IT.
- Appendix 8 - *MIŠKAS* – THE INSTRUMENT AND HOW TO PLAY IT.

APPENDIX 2

THE TWO GENERIC INSTRUMENTS

THE FIRST GENERIC INSTRUMENT

(The first generic instrument used in *NINE STATIONS* – I and in *UNDER THE EIFFEL TOWER* - I.)



```
        1 & 2                3 & 4                5 & 6
        synth:               dir:                 numSynths:
        bufPlayer            01-9F-SS             1
shift   rq:                  attack:              bend:           shift
        0.001                10                   1
ctrl    startRate:           rateStep:                            ctrl
        1                    0.03
        1 & 2                3 & 4                5 & 6
```

Keys and modifiers:

- 1 & 2 changes current synth. Available synths are: bufPlayer, fadeInBendPlayer, filter, lineFilterTW, lineFilterTT and echo. See descriptions below.
- 3 & 4 changes sound directory.
- 5 & 6 changes numSynths, i.e. number of simultaneous synths.
- Shift + 1 & 2 RQ - changes current RQ value
- Shift + 3 & 4 attack, changes the attack.
- Shift + 5 & 6 bend, changes the pitch bend. 1 means no bend.
- Ctrl + 1 & 2 startRate - the playback rate of the keyboards leftmost and lowest (on the same row as the space bar) key.
- Ctrl + 3 & 4 rateStep. How much higher the play back rate will be on the next key to the right.

Description of the synths used:

- bufPlayer – a simple bufPlayer that plays a sound at a desired rate.
- fadeInBendPlayer – A bufPlayer that fades in the sound and changes the speed during playback (like a pitch bend). The amount of both effects is chosen by the user. Both can be no effect at all.
- filter – A Band Pass Filter (BPF).
- lineFilterTW – A BPF with changing RQ from a low value to a higher value which makes more of the sound audible.

- lineFilterTT – The opposite of above.
- echo – An echo effect in the form of a Tdef that plays a sound again and again with a diminishing amplitude and a short pause between the events.

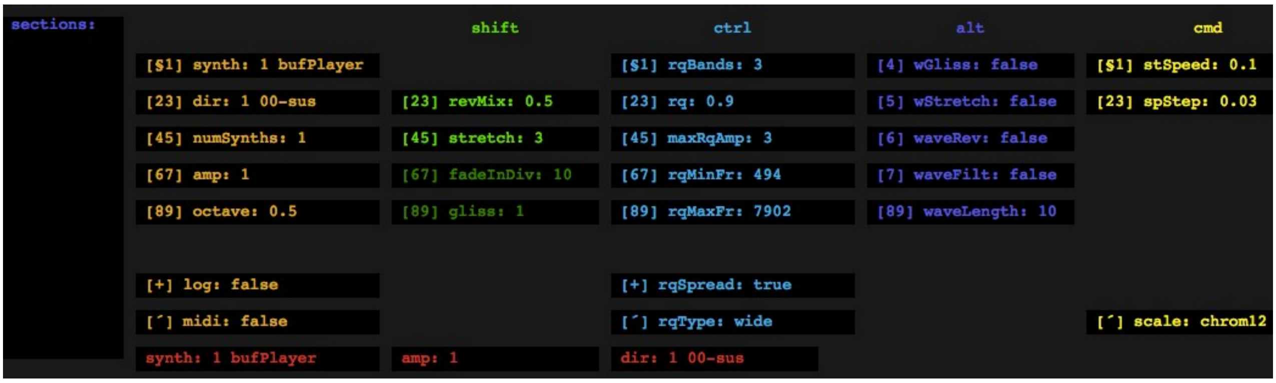
Alternatives for startRate and rateStep:

- startRates = 0.25, 0.5, 0.75, 1, 1.25.
- rateSteps = 0.001, 0.003, 0.006, 0.01, 0.03.

I usually use startRate 1 and rateStep 0.03, which gives me a scale that begins like this (presuming that I use a sound with fundamental 100 Hz): 100, 103, 106, 109 etc. Compared to a 12-ET which would be 103.826 (G#), 110 (A), 116.541 (A#), 123.471 (B2).

THE SECOND GENERIC INSTRUMENT

This instrument is a bit different from the others. It has the modifier to use as a headline and the colour coding follows the modifier. The number key to press is written out before the buttons name.



sections:		shift	ctrl	alt	cmd
	[$1] synth: 1 bufPlayer		[$1] rqBands: 3	[4] wGliss: false	[$1] stSpeed: 0.1
	[23] dir: 1 00-sus	[23] revMix: 0.5	[23] rq: 0.9	[5] wStretch: false	[23] spStep: 0.03
	[45] numSynths: 1	[45] stretch: 3	[45] maxRqAmp: 3	[6] waveRev: false	
	[67] amp: 1	[67] fadeInDiv: 10	[67] rqMinFr: 494	[7] waveFilt: false	
	[89] octave: 0.5	[89] gliss: 1	[89] rqMaxFr: 7902	[89] waveLength: 10	
	[+] log: false		[+] rqSpread: true		
	[´] midi: false		[´] rqType: wide		[´] scale: chrom12
	synth: 1 bufPlayer	amp: 1	dir: 1 00-sus		

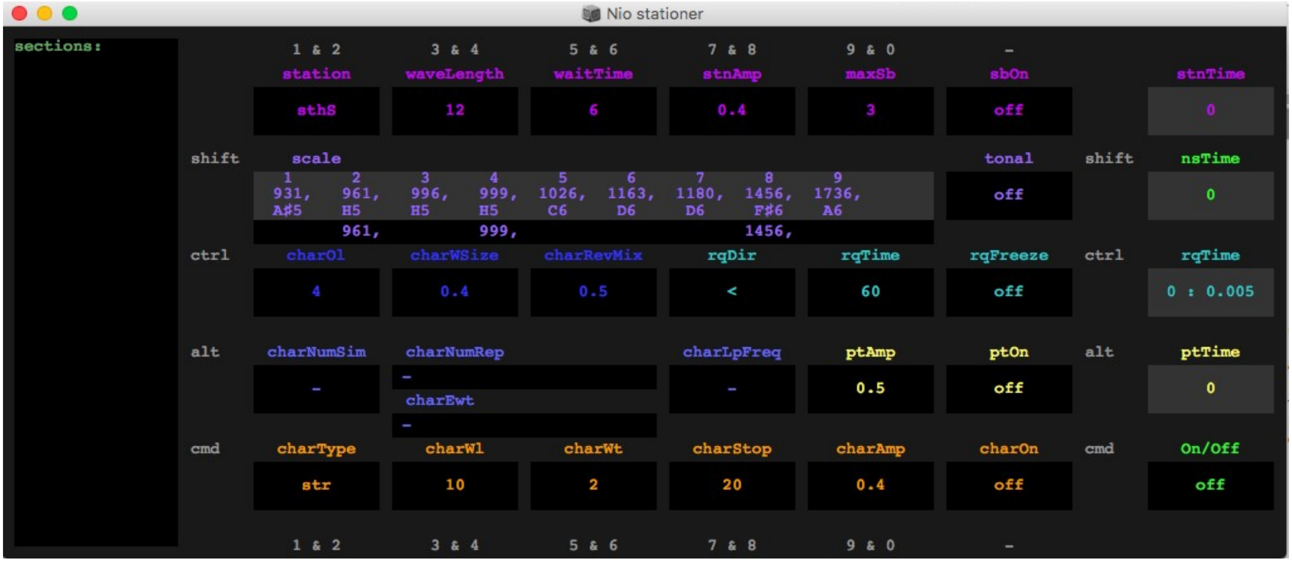
Description of the instrument from top left.

- sections: a list of the sections that is currently running. The sections are started manually by running rows of code.
- row 1: description of which modifier to use. They are colour coded to make playing easier.
- column 1 (no modifier):
  - synth: which synth to use.
  - dir: which sounds to use.
  - numSynths: How many of the chosen synth to play simultaneously.
  - amp: Amplitude.
  - octave: transposition.
  - log: whether to use SuperColliders History or not.
  - midi: whether to use an external midi keyboard or not.

- column 2 (shift):
    - revMix:this and the following buttons controls arguments to the warp1 synth. This controls revMix.
    - stretch: as above. Controls stretch amount.
    - fadeInDiv. The same function but another name for Attack. Works like the Attack value in *NINE STATIONS* - I.
    - gliss. Another name for pitch bend.
- column 3 (ctrl):
    - rqBands: how many simultaneous filters to use.
    - rq: the RQ value used by BPF filters.
    - rqMaxAmp: The maximum amplitude used by the filter synths.
    - rqMinFreq: The lowest frequency the filters will use.
    - rqMaxFreq: The highest frequency the filters will use.
    - rqSpread: Whether to send all filter frequencies to the same speaker or different speakers.
    - rqType: Different sound types have different mappings between amplitude and RQ. I have two types; thin and wide. The thin type has higher amplitudes than the wide type.
- column 4 (alt):
    - wGliss: Pitch bend effect for running sections.
    - wStretch: Time stretch effect for running sections.
    - waveRev: Reverb effect for running sections.
    - waveFilt: Filter effect for running sections.
    - waveLength: a running sections length in seconds.
- column 5 (cmd)
    - stSpeed: The same function as startRate in *NINE STATIONS* - I.
    - spStep: The same function as rateStep in *NINE STATIONS* - I.
    - scale: which scale or tuning to use on the computers keyboard.
- The red row at the bottom refers to an external midi keyboard. If I connect an external midi keyboard these three buttons control which synth to use, the amplitude and what sounds to use.

APPENDIX 3

NINE STATIONS - II – THE INSTRUMENT AND HOW TO PLAY IT



```
Nio stationer

sections:      1 & 2        3 & 4        5 & 6        7 & 8        9 & 0        -
               station      waveLength   waitTime     stnAmp       maxSb        sbOn                 stnTime
               sthS         12           6            0.4          3            off                  0

shift          scale                                                            tonal        shift   nsTime
                 1      2      3      4      5      6      7      8      9
               931,   961,   996,   999,   1026,  1163,  1180,  1456,  1736,  off                  0
               A#5    H5     H5     H5     C6     D6     D6     F#6    A6
                      961,          999,                         1456,

ctrl           charOl       charWSize    charRevMix   rqDir        rqTime       rqFreeze     ctrl    rqTime
               4            0.4          0.5          <            60           off                  0 : 0.005

alt            charNumSim   charNumRep                charLpFreq   ptAmp        ptOn         alt     ptTime
               -            -                         -            0.5          off                  0
                            charEwt
                            -

cmd            charType     charWl       charWt       charStop     charAmp      charOn       cmd     On/Off
               str          10           2            20           0.4          off                  off

               1 & 2        3 & 4        5 & 6        7 & 8        9 & 0        -
```

Description of the instrument from top left.

- sections: a list of the sections that are currently running and - when one has chosen to stop a section, a counter that says for how much longer it will run. A section is a combination of certain sounds and a synth. The combination has a name and default values that can be changed by the user.
- row 1: numbers indicating which keys to use to change the value of the buttons in the same column.
- station: which stations sounds that are currently played. The selected station also decides if some of the other buttons are available or not, for instance, passing trains are not available for all stations.
- waveLength: the length of the waves, i.e. the length of the played sound, fading in and out and therefore called a wave.
- waitTime: number of seconds to wait before starting a new wave.
- stnAmp: amplitude of the station waves.
- maxSb: maximum number of - what I call sideBands (octaves transpositions of already chosen frequencies). The maximum number of additional filter frequencies. Always a multiple or division by two in this piece.
- sbOn: whether to use sideBands or not.
- stnTime: information field showing for how long the current station have been active.
- scale: the current stations nine frequency scale. All frequencies can be turned on or off by pressing shift and it's numerical value. (All but one, there will always be at least one frequency running.)

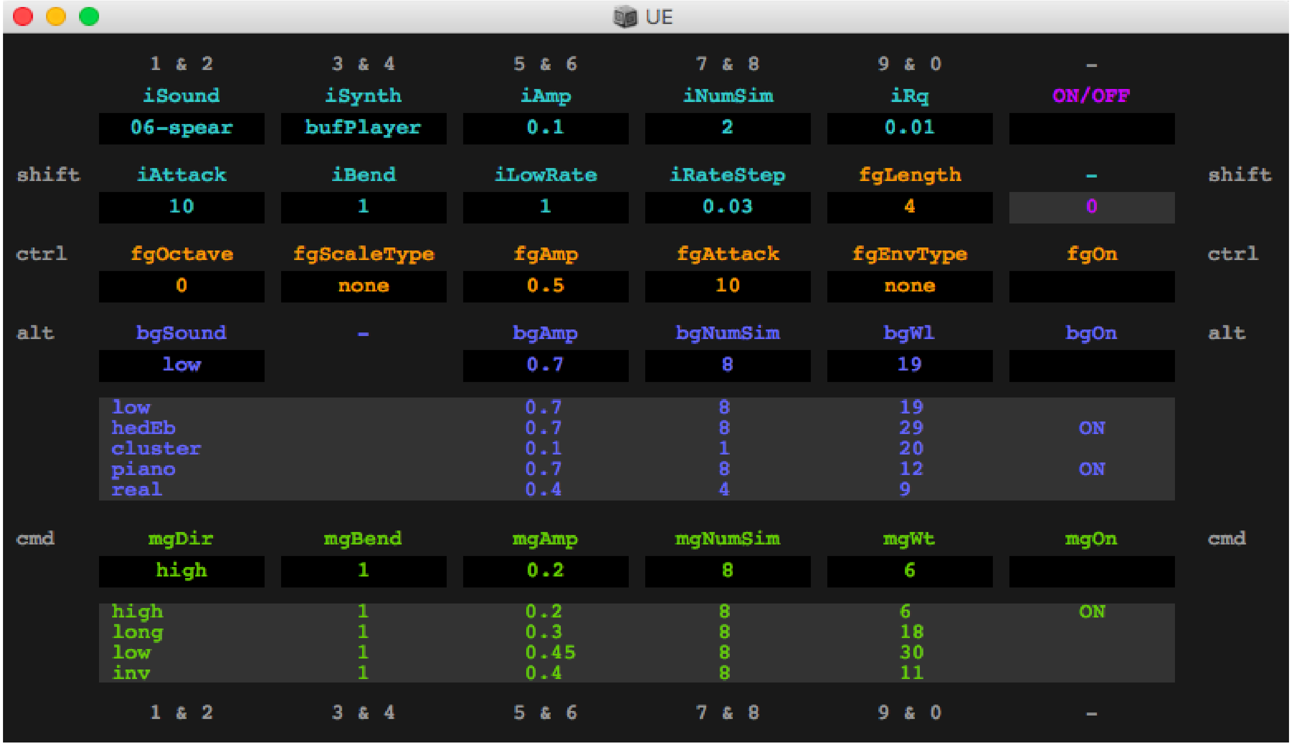
- tonal: whether we should force the frequencies into frequencies that belong to our 12-ET scale or not. If we choose tonal=on the listed frequencies will be moved to the nearest "tonal" frequency.
- charOl: this and the two following buttons control arguments to the warp1 synth. charOl is numberOfOverlaps.
- charWSize: as above, controls windowSize
- charRevMix: as above, controls revMix
- rqDir: controls in which direction the rq-counter is running. < means to wider, > means to narrower, - means that it stands still.
- rqTime: a changeable value that decides for how many seconds the rq-counter will take when going from the widest rq to the narrowest, and vice versa.
- rqFreeze: the same as - above. Don't move the rq. Let it stay where it is.
- rqTime: shows for how long the current rq-cycle has been running.
- charNumSim: numSim, “number of simultaneous” controls how many synths of the chosen sort we play at the same time.
- charNumRep: For the percussive effect. It means “how many repetitions”.
- charEwt: As above, the wait time, i.e. how long pause we want between the repetitions.
- charLpFreq: As above, a Low Pass filter frequency.
- ptAmp: amplitude of a passing train.
- ptOn: start a passing train.
- ptTime: show for how much longer a passing train will run.
- charType: toggle button that lists all the different char types the currently chosen station has.
- charWl: as waveLength above but for the chosen charType.
- charWt: as waitTime above but for the chosen charType.
- charStop: how many seconds it will take for this charType to fade out and stop running.
- charAmp: amplitude for the current charType.
- charOn: whether this charType is running or not. When it is running, pressing it again will stop the section. Currently running charTypes is listed in the sections button. When you want to stop a charType section you have to choose the right charType value in the charType button and press cmd + -. This is a little distressing when actually playing the instrument. I would want an easier way to stop a section and have thought of giving each section a letter and stopping it by pressing, for instance cmd and the letter.
- on/off: whether NS is on or not.

HOW TO PLAY *NINE STATIONS* - II

- Turn *NINE STATIONS* - II on by pressing the = sign.
- *NINE STATIONS* - II starts automatically with the chosen station, which default is SS. To change station press keys 1 or 2.
- Change which of the nine frequencies to play by pressing shift and the number above the frequency.
- Choose CH sound by pressing cmd and 1 or 2. Start it by pressing cmd + the minus sign. Stop it by choosing the same CH again and press cmd + -. The CH sounds you have running is listed in the sections button to the left in the GUI. You can but should not restart a CH type before it has disappeared from the sections list. If you do the first instance becomes orphaned and difficult to stop. This should of course be prevented in the next version of the instrument.

APPENDIX 4

UNDER THE EIFFEL TOWER - II - THE INSTRUMENT AND HOW TO PLAY IT



	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	
	iSound	iSynth	iAmp	iNumSim	iRq	ON/OFF	
	06-spear	bufPlayer	0.1	2	0.01		
shift	iAttack	iBend	iLowRate	iRateStep	fgLength	-	shift
	10	1	1	0.03	4	0	
ctrl	fgOctave	fgScaleType	fgAmp	fgAttack	fgEnvType	fgOn	ctrl
	0	none	0.5	10	none		
alt	bgSound	-	bgAmp	bgNumSim	bgWl	bgOn	alt
	low		0.7	8	19		
	low		0.7	8	19		
	hedEb		0.7	8	29	ON	
	cluster		0.1	1	20		
	piano		0.7	8	12	ON	
	real		0.4	4	9		
cmd	mgDir	mgBend	mgAmp	mgNumSim	mgWt	mgOn	cmd
	high	1	0.2	8	6		
	high	1	0.2	8	6	ON	
	long	1	0.3	8	18		
	low	1	0.45	8	30		
	inv	1	0.4	8	11		
	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	

Description of the instrument from top left.

Turquoise buttons starting with the letter i (as in Instrument) are for controlling the playing on the computer keyboard.

- iSound – which sound directory to use.
- iSynth – which synth to use.
- iAmp – the amplitude.
- iNumSim – number of simultaneous synths.
- iRq – the rq value (when using a filter synth).
- iAttack – the sounds attack.
- iBend – pitch bend.
- iLowRate – controls the rate of the leftmost key on the row above the spacebar.
- IRateStep - How much higher the play back rate will be on the next key to the right of the leftmost key.

Orange buttons starting with letters fg (foreground) controls the playing of the partial analysis.

- fgLength – only play partials that are as long or longer that fgLength. A high fgLength value makes the sound less dense.

- fgOctave – 0 means the unaltered pitch from the partial analysis. A higher number means to transpose the pitches in the partial analysis to the specified octave. Seven is the highest octave.
- fgScaleType – this can be none, ET12 or siren. Siren is the pitches from the emergency vehicle in the December 9 recording.
- fgAmp – the amplitude.
- fgAttack – the sounds attack. Has no effect unless “fadeIn” is chosen below.
- fgEnvType – this can be none, fadeIn or env. FadeIn removes the attack from the sounds. How much is specified by the attack value above. Env means to follow the partials envelop. If env is not chosen, only the partials start values are used.
- fgOn – whether to play the partial analysis data or not.

Five blue buttons with names starting with the letters bg (background) and a list of sounds. These are the December 9 files that were also used in *UNDER THE EIFFEL TOWER* - I.

- bgSound – which sound.
- bgAmp – the amplitude.
- bgNumSim – how many simultaneous synths.
- bgWl – wave length. How long portion of the sound that will be played before it fades out and then back in again.
- bgOn – whether to play this December 9 file or not.

Six green buttons with names starting with the letters mg (middle ground) and a list of directories. Controls the recurring BEDEb and low B sounds.

- mgDir – which sound directory.
- mgBend – pitch bend.
- mgAmp – the amplitude.
- mgNumSim – how many simultaneous synths.
- mgWt – waitTime. Number of seconds to wait before playing the next sound.
- mgOn – whether to play sounds from this directory or not.

HOW TO PLAY *UNDER THE EIFFEL TOWER -* II

Without even starting UNDER THE EIFFEL TOWER - II you can play on the computer keyboard.

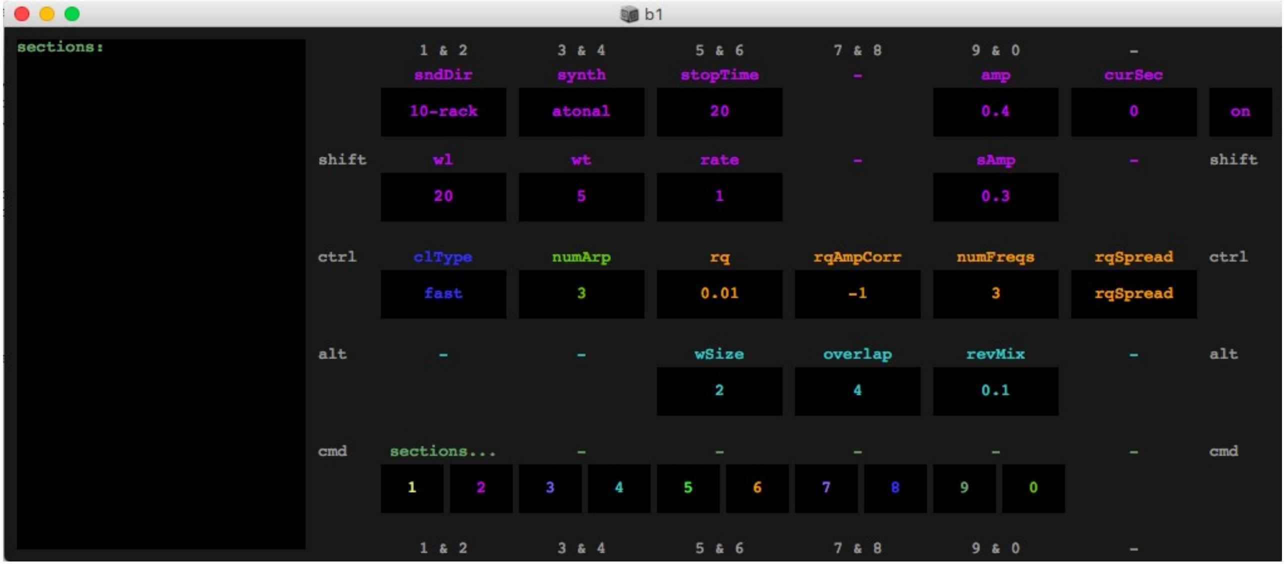
Change the sound directory with characters 1 and 2. Change play method / synth with characters 3 and 4. All turquoise buttons with names beginning with “i” can be used.

Start *UNDER THE EIFFEL TOWER* - II by pressing the - character. It will start the sounds marked with ON in the blue and green areas of the GUI. You can change this by pressing alt and 1 or 2 to choose the sound you want. When a sound is chosen you can change its values according to the labels.

Stop *UNDER THE EIFFEL TOWER* - II by pressing the - character again.

APPENDIX 5

BUS NO. 1 – II - THE INSTRUMENT AND HOW TO PLAY IT



b1

sections:

	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	
	sndDir	synth	stopTime	-	amp	curSec	
	10-rack	atonal	20		0.4	0	on
shift	wl	wt	rate	-	sAmp	-	shift
	20	5	1		0.3		
ctrl	clType	numArp	rq	rqAmpCorr	numFreqs	rqSpread	ctrl
	fast	3	0.01	-1	3	rqSpread	
alt	-	-	wSize	overlap	revMix	-	alt
			2	4	0.1		
cmd	sections...	-	-	-	-	-	cmd
	1 2	3 4	5 6	7 8	9 0		
	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	

The first left column shows the sections. When a section is playing its button gets highlighted and it is listed in this sections area.

Next to the sections column, from top left:

- sndDir – which sounds this section will use.
- synth – which synth this section will use.
- stopTime – for how long a section will fade out after it has been turned off.
- amp – the instrument amplitude.
- curSec – current section – the section number that all buttons control and the one that will be turned on when pressing -, or turned off if it is already playing.
- on – turn *BUS NO. 1* - II on or off.

Row two from left to right:

- wl – waveLength – how long part of the sound to play.
- wt – waitTime – how long to wait before playing the sound again.
- rate – the playback rate.
- sAmp – the sections amplitude.

Row three and four:

- These all belong to the *AirIac* instrument. They could used in *BUS NO. 1* - II but I haven't done so in the submitted version of *BUS NO. 1* - II and I wont describe them here.

The bottom row shows the ten preprepared sections. You select one of them by pressing cmd and its number.

HOW TO PLAY *BUS NO. 1* - II

Start *BUS NO. 1* - II by pressing the ´ character. Serially filtered bus sounds will start playing in eight channels. You can’t control these bus-sounds. They will play for the duration of the piece, 9.12. (This should be changed in the future.)

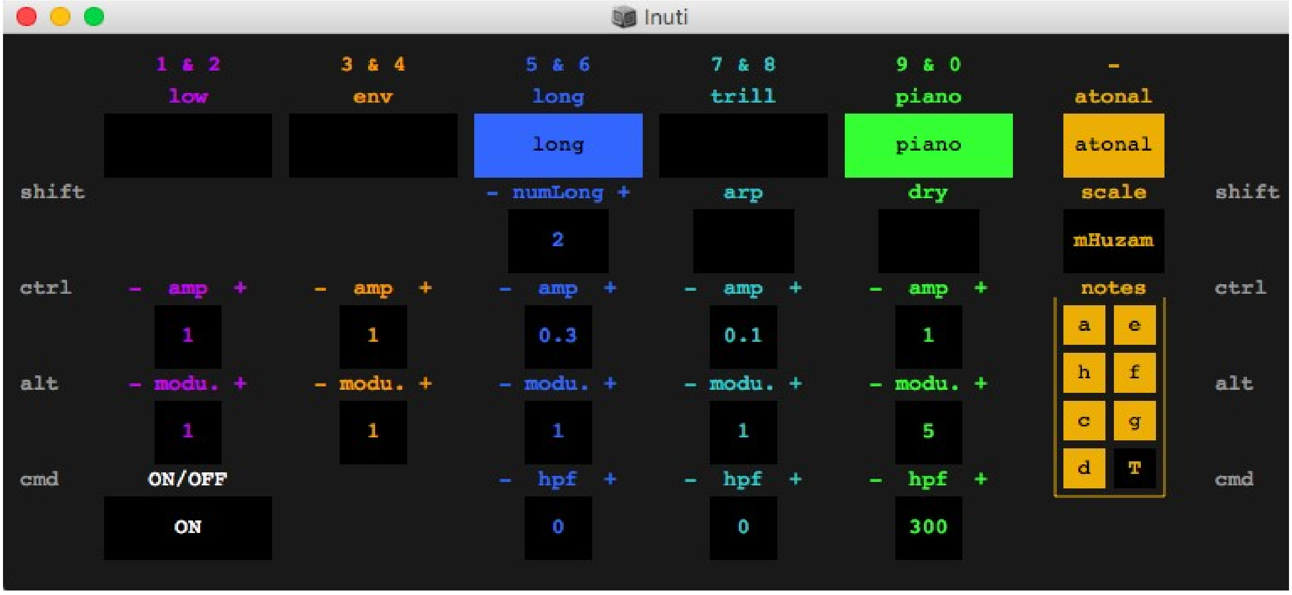
Next, choose a section by pressing cmd + the sections number. You start the section by pressing the - character. You will see it listed in the sections area of the GUI. You can play ten sections at the same time.

To stop a section. Choose it by pressing cmd and it’s number and press the - character again. You will see that its listing in the sections area will start counting down. This is the “stopTime”. You can change its value in the GUI. When the counting has finished it will disappear from the sections area.

To change a sections amplitude press shift and 9 to diminish and shift and 0 to increase.

APPENDIX 6

INUTI – THE INSTRUMENT AND HOW TO PLAY IT



In *INUTI* there are two types of sounds, the recorded piano strings and time stretched 60 seconds long piano sounds. Only the *long* play type use the long sounds. The five play types correspond to the five selections of partials. Low, env, long, trill and piano have one column and one colour each in the GUI. Long has an extra button: numLong (called numSim in other instruments). Trill has two play types: Trill and Arpeggio. Play type piano has a dry option to make the notes shorter.

Starting from top left:

- Keys 1 and 2 - *low* – the selection of the lowest partials.
  - Without modifier. Turn *low* on or off.
  - With ctrl – controls the amplitude. 1 decreases, 2 increases.
  - With alt – controls the modulus value. 1 decreases, 2 increases.
  - With cmd. The white button is the main on/off button, and has nothing to do with *low* as is indicated by its different colour.
- Keys 3 and 4 – *env* – the selection of the longest partials.
  - Without modifier. Turn *env* on or off.
  - With ctrl – controls the amplitude. 3 decreases, 4 increases.
  - With alt – controls the modulus value. 3 decreases, 4 increases.
- Keys 5 and 6 – *long* partials. The sounds for the *long* play type are different from the rest. They are time stretched piano tones about 60 seconds long.
  - Without modifier. Turn *long* on or off.

    - With shift – controls how many *long* notes the instrument should play simultaneously.
    - With ctrl – controls the amplitude. 5 decreases, 6 increases.
    - With alt – controls the modulus value. 5 decreases, 6 increases.
    - With cmd – controls the HPF. 5 decreases, 6 increases.
- Keys 7 and 8 – *trill* partials.
    - Without modifier. Turn *trill* on or off.
    - With shift – Changes the *trill* to an *arp* (arpeggio)
    - With ctrl – controls the amplitude. 7 decreases, 8 increases.
    - With alt – controls the modulus value. 7 decreases, 8 increases.
    - With cmd – controls the HPF. 7 decreases, 8 increases.
- Keys 9 and 0 – *piano* partials.
    - Without modifier. Turn *piano* on or off.
    - With shift – Makes the *piano* tones dry (short).
    - With ctrl – controls the amplitude. 9 decreases, 0 increases.
    - With alt – controls the modulus value. 9 decreases, 0 increases.
    - With cmd – controls the HPF. 9 decreases, 0 increases.
- The - sign toggles between atonal and a scale.
    - with shift the user can choose which scale to use.
- The notes can be turned on or off by pressing ctrl and the name of the note. The button T means Tutti, i.e, all.

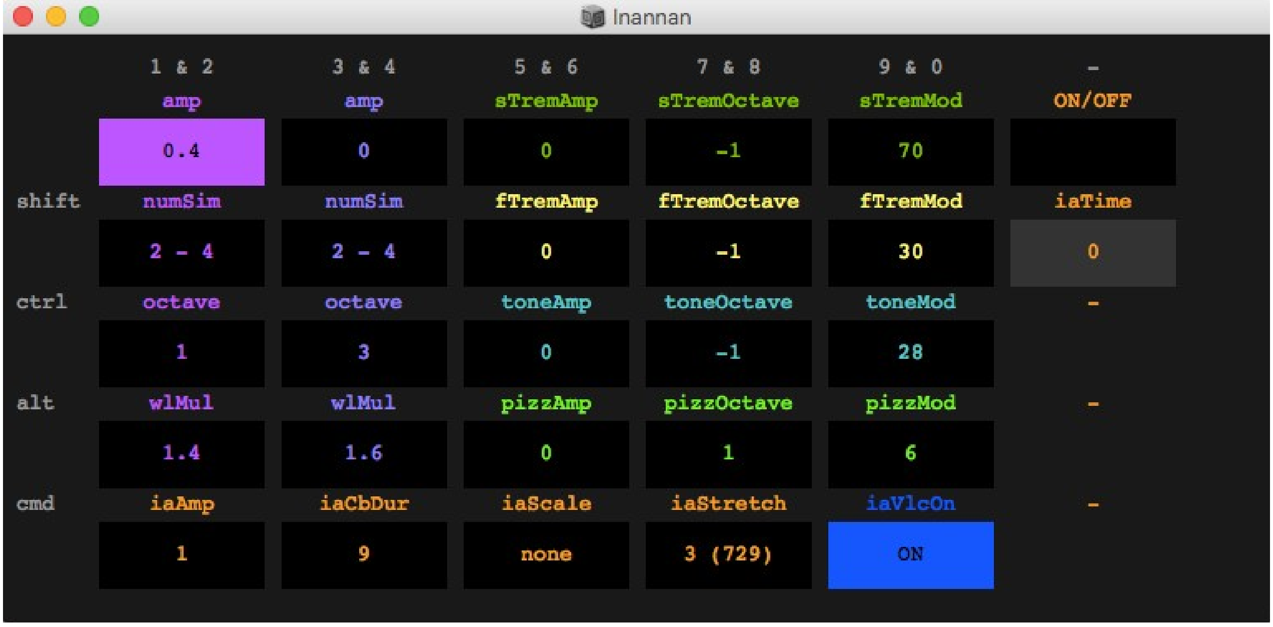
HOW TO PLAY *INUTI*

In *INUTI* the instrument loops over the data in the SDIF file. For each iteration it checks which of the five play methods is enabled, whether it has a partial to play and whether its modulus value says that it should play a sound.

The user simply turns these play methods on or off, adjusts the amp, modulus and HPF. The user can also control the tonality by choosing atonal, a scale or individual notes.

APPENDIX 7

INANNAN – THE INSTRUMENT AND HOW TO PLAY IT



	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-
	amp	amp	sTremAmp	sTremOctave	sTremMod	ON/OFF
	0.4	0	0	-1	70	
shift	numSim	numSim	fTremAmp	fTremOctave	fTremMod	iaTime
	2 - 4	2 - 4	0	-1	30	0
ctrl	octave	octave	toneAmp	toneOctave	toneMod	-
	1	3	0	-1	28	
alt	wlMul	wlMul	pizzAmp	pizzOctave	pizzMod	-
	1.4	1.6	0	1	6	
cmd	iaAmp	iaCbDur	iaScale	iaStretch	iaVlcOn	-
	1	9	none	3 (729)	ON	

INANNAN has two "layers". One is the playing of the sliced sound files, the other is cello sounds. In the GUI, the two leftmost columns control the sliced files.

The colours show what belongs together:

- Cerise column: Sliced files played with the wave synth.
- Violet column: Sliced files played with the stretch synth.
- Apple green row: Slow cello tremolo.
- Yellow row: Fast cello tremolos.
- Turquoise row: Cello tones.
- Green row: Cello pizzicatos.
- Orange row and column: Controls all of *INANNAN*.
- Blue button: Turns the automatically triggered cello sounds on or off.

The two left columns control the playing of the slices. The leftmost is the bufplayer/wave synth and the one to the right is the stretch player. The both have numSim (how many simultaneous synths), octave and wlMul (a float which a sounds length gets multiplied by, to avoid silences between the tones). The other buttons should be read from top to bottom. First sTrem (slow tremolo), then fTrem (fast tremolo) tone and pizzicato. These all have buttons for amplitude, octave and modulus. The bottom row is:

- iaAmp – *INANNAN*'s amplitude

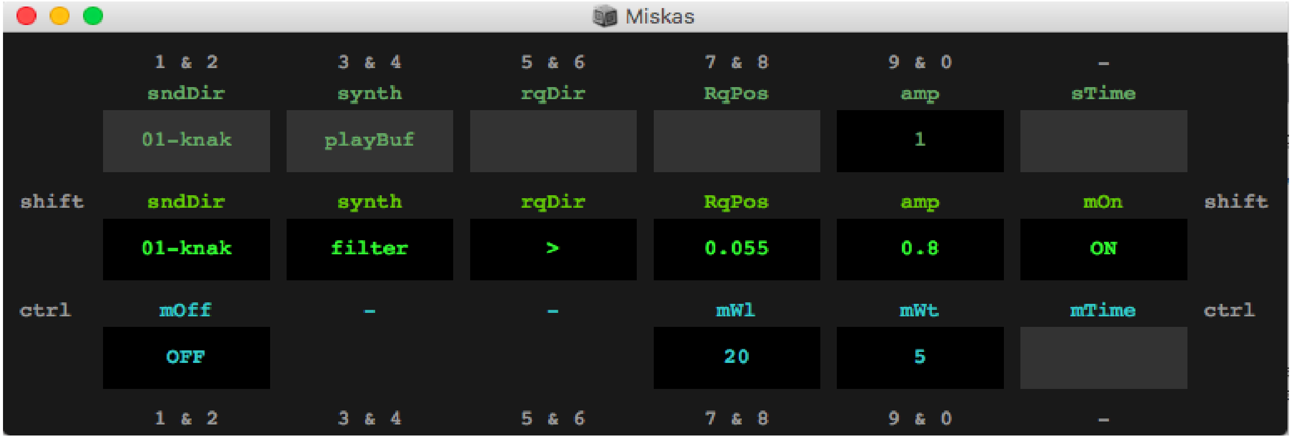
- iaCbDur – the duration in seconds (from the SDIF file) that (if the blue button IaVlcOn is ON) triggers the cello sounds.
- iaScale – which scale to use, or no scale at all.
- iaStretch – the stretch variable that all of *INANNAN* gets stretched by. If set to 1 *INANNAN* will be 4.03 long (like the original cello melody).
- iaVlcOn – whether or not to use the automatically triggered cello sounds.

HOW TO PLAY *INANNAN*

- Start *INANNAN* by pressing the - key.
- The cello sounds will start automatically if iaVlcOn is true and if a value from the SDIF file is higher than iaCbDur. IaVlcOn is true by default.
- amp 0 means off. Amps higher than 0 is on.
- Turn the different slice-playing synths on or off using the amp. Change the different GUI controllers to see how it sounds.
- Press the - sign when you are done.

APPENDIX 8

MIŠKAS – THE INSTRUMENT AND HOW TO PLAY IT



	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	
	sndDir	synth	rqDir	RqPos	amp	sTime	
	01-knak	playBuf			1		
shift	sndDir	synth	rqDir	RqPos	amp	mOn	shift
	01-knak	filter	>	0.055	0.8	ON	
ctrl	mOff	-	-	mWl	mWt	mTime	ctrl
	OFF			20	5		
	1 & 2	3 & 4	5 & 6	7 & 8	9 & 0	-	

In *MIŠKAS* you run “sections”. A section is a sound directory, a synth, an amplitude, if the synth is a filter it is also an RQ-direction and an RQ-start value. You can only run one section at a time. In the top row in the GUI you see what is currently running (or will run if you haven’t started *MIŠKAS* yet). The second row shows the values for the next section. These values can be changed until you start this next section (by pressing shift - the plus sign).

Top row starting from the left: (currently used values or the first ones that will be used when you start *MIŠKAS*.)

- sndDir - which sounds are used.
- synth - which synth is used- Available synths are:
  - playBuf – plain sound file player.
  - filter – a player that filters the sound according to the chosen RQ value.
  - bendFilter – the same as filter but with a randomly chosen pitch bend amount. You can not control the pitch bend.
- rqDir – the direction of the RQ-value. It can be:
  - - no change.
  - > value changes to a smaller value making the sound thinner.
  - < value changes to a larger number making the sound thicker.
- rqPos - from where to start in the rq-list (for RQ-lists, see below).
- amp - this sections amplitude.
- sTime - shows for how long this section has been running.

Middle row: (Values for the next section. These values will replace the currently running values when you press shift + -.)

- sndDir - which sounds to use.
- synth - which synth to use.
- rqDir - whether the RQ value should become smaller (thinner sound) or larger (thicker sound).
- rqPos - from where to start in the rq-list.
- amp – this sections amplitude.
- mOn – Pressing shift - the plus sign starts *MIŠKAS* and changes to the next section.

Bottom row: Except for the OFF button, these values can only be changed while *MIŠKAS* is off.

- OFF - turn MIŠKAS off by pressing ctrl + 1 (or 2).
- mWl - the waveLength, i.e. for how long time in seconds a synth will play a soundfile.
- mWt - the waitTime, i.e.for how long *Miškas* will wait before starting a new synth.
- mTime - shows for how long *Miškas* has been running.

HOW TO PLAY *MIŠKAS*

- In the GUI, press shift + -. You will hear a crackling sound.
- Choose values for the next section and press shift + the minus sign when you want to switch to the next section.
- Press ctrl + 1 when you are done.

MIŠKAS RQ-LISTS

MIŠKAS uses two different RQ-lists depending on the sounds character.

For the crackling sound this list is used:

[0.055, 0.05, 0.045, 0.04, 0.035, 0.03, 0.025, 0.02, 0.015, 0.01, 0.0095, 0.009, 0.0085, 0.008, 0.0075, 0.007, 0.0065, 0.006, 0.0055, 0.005, 0.0045, 0.004, 0.0035, 0.003, 0.0025, 0.002, 0.0015, 0.001, 0.00095, 0.0009, 0.00085, 0.0008, 0.00075, 0.0007, 0.00065, 0.0006, 0.00055, 0.0005, 0.00045, 0.0004]

The crackling sounds are thin frequency wise and it does not sound “good” to use the same RQ values for all sound types.

For the other sounds this list is used:

[0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5, 0.45, 0.4, 0.35, 0.3, 0.25, 0.2, 0.15, 0.1, 0.095, 0.09, 0.085, 0.08, 0.075, 0.07, 0.065, 0.06, 0.055, 0.05, 0.045, 0.04, 0.035, 0.03, 0.025, 0.02, 0.015, 0.01, 0.0095, 0.009, 0.0085, 0.008, 0.0075, 0.007, 0.0065, 0.006, 0.0055, 0.005, 0.0045, 0.004, 0.0035, 0.003, 0.0025, 0.002, 0.0015, 0.001]

APPENDIX 9

HhCLASSES

These are two utility classes that I use for handling sound directories and SDIF data.

HhSounds has functions:

- loadSounds(server, topDir) – it takes a path to a directory as an argument and loads the sounds in its subdirectories. It loads everything that ends with .aif or .aiff. It prints a report of what it has loaded in the post window.
- freeSounds() – frees the loaded sounds.
- getSoundDirs() - returns the loaded sound directories.
- getDirNames() - returns the names of the loaded directories.
- getSound(dirNo, fileName) – returns the buffer fileName in the specified directory.
- getFileNameFromBufnum(dirNo, bufnum) – self explanatory.
- listSounds – lists all loaded sounds.
- listSoundsInDir(dirNo) – lists all sounds in the specified directory.
- listDirs – lists the names of the loaded directories.
- loadSoundDirs(server, topDir) – like loadSounds but this function expects that the sound directories have files with names of format xx-nnn-nn.aif. For example p-110-01.aif. In this example p is just a name, 110 is a frequency and 01 is a counting number. If there is more then one file with frequency 110 this one would be called p-110-02.aif. This function loads the sound files and stores the frequency information in the two arrays bufnum2freq and freq2bufnums so you can find a sound via frequency.
- getSoundsNFreqs – returns the two arrays bufnum2freq and freq2bufnums mentioned above.

HhSdif has functions:

- readTextTrcSdif(sdifFile, channel) – reads an SDIF file with a partial analysis exported from AudioSculpt. This function reads the SDIF in txt format.
- readBinaryTrcSdif(sdifFile, channel) – reads an SDIF file with a partial analysis exported from AudioSculpt. This function reads the SDIF in binary format.
- readTextF0Sdif(sdifFile) - reads an SDIF file with a fundamental analysis exported from AudioSculpt. This function reads the SDIF in txt format.
- readBinaryF0Sdif(sdifFile)- reads an SDIF file with a fundamental analysis exported from AudioSculpt. This function reads the SDIF in binary format.

- sortPartialKeys(partials) – returns a list with the partials sorted in number order.
- orderByPartialNo(sdif) – returns the SDIF data ordered by the partials number. The read functions above returns the data ordered by time.
- choosePartials(partials, modulus, minLength, maxLength, minFreq, maxFreq, minAmp, maxAmp) – selects a some of all partials.
- timeGrid(partials, offset, secs) – selects some of all partials according to a time grid. For example one each ninth second.
- envArrays(inPartials, soundData, attack, delay, maxFreq, stretch, ampAdd) – return a list with data suitable for making envelopes according to the partials data.

APPENDIX 10

PORTFOLIO CONTENT

In this portfolio there is a directory called images. This directory contains two directories called guis and sonograms. The guis directory contain big images of all guis described in this text. The directory sonograms contain big, uncropped images of all sonograms described in this text.

- guis:
  - 01-NS1-gui.png (*NINE STATIONS* - I*)*
  - 01-NS2-gui.png (*NINE STATIONS* - II*)*
  - 02-UE2.gui.png (*UNDER THE EIFFEL TOWER* - II)
  - 03-B11-gui.png (*BUS NO. 1* - I)
  - 03-B12-gui.png (*BUS NO. 1* - II)
  - 04-IN-gui.png (*INUTI*)
  - 05-IA-gui.png (*INANNAN*)
  - 06-MI-gui.png (*MIŠKAS*)
- sonograms:
  - 01-NS1-mono.png (*NINE STATIONS* - I)
  - 01-NS2-mono.png (*NINE STATIONS* - II)
  - 02-UE1-mono.png (*UNDER THE EIFFEL TOWER* - I)
  - 02-UE2-mono.jpg (*UNDER THE EIFFEL TOWER* - II)
  - 03-B11-mono.jpg (*BUS NO. 1* - I)
  - 03-B12-mono.jpg (*BUS NO. 1* - II)
  - 04-IN-mono.png (*INUTI*)
  - 05-IA-mono.png (*INANNAN*)
  - 06-MI-mono.png (*MIŠKAS*)

There is also a directory called instruments. This directory contains the SuperCollider code discussed in this text.

- instruments:
  - HH-classes.tgz
  - HH-NS.tgz (*NINE STATIONS* II)
  - HH-UE.tgz (*UNDER THE EIFFEL TOWER* - II)

    - HH-B1.tgz (*BUS NO. 1* II)
    - HH-IN.tgz (*INUTI*)
    - HH-IA.tgz (*INANNAN*)
    - HH-MI.tgz (*MIŠKAS*)

Finally, there is a directory called rp which contains my RESEARCH PROPOSAL sound examples.

- 1-saxOriginal.aif
- 2-saxCut.aif
- 3-saxBreath.aif
- 4-MellanrumMonoExcerpt.aif
- 5-laundryRackOriginal.aif
- 6-laundryRackString.aif
- 7-laundryRackPing.aif
- 8-laundryRackSheet.aif
- 9-spoons.aif
- 10-spoonChoir.aif

APPENDIX 11

RESEARCH PROPOSAL

1. An outline of your chosen topic.

I have two main areas of interest. One is to unmask previously masked features in recordings that I have done, the other is an instrument/composition tool that I am working on.

Here are some examples of the first area of interest:

For my piece Mellanrum I recorded three musicians playing their instrument. I wasn't interested in what they played but in how they sounded while doing it. While editing I removed all loud instrument sounds and kept only what was left – the player himself and very soft instrument sounds that were not heard before the removal of the loud sounds.

Sound examples:

- 1-saxOriginal.aif – part of the original recording

- 2-saxCut.aif – after removing the loud sounds

- 3-saxBreath.aif – montage of breath sounds

- 4-MellanrumMonoExcerpt.aif – example of using the above in the piece

For another piece; December 9 - triptych part I I recorded when a sheet was drawn over a laundry rack. It makes a loud screeching sound. I filtered the sound repeatedly in AudioSculpt - I removed the loudest frequencies and found frequencies that had previously been masked by the louder ones. I also separated the string sound from the sheet sound and used both in the piece.

Sound examples:

- 5-laundryRackOriginal.aif – part of the original recording

- 6-laundryRackString.aif – filtered to a string like sound

- 7-laundryRackPing.aif – filtered even more

- 8-laundryRackSheet.aif – string sound removed and the drawn sheet heard more clearly

The second area of interest – my instrument/composition tool.

This area is not very different from the first. I am working on a tool in which I try to find as many ways as possible of playing sound files. I can vary such things as speed, tuning and envelopes. For example, in the same piece as above; December 9 - triptych part II I used a recording of spoons being hit against one another and discovered that if I played the sound file very slowly and many at the same time with a very small difference in speed it started to sound like a choir.

Sound examples:• 9-spoons.aif – part of the original recording

- 10-spoonChoir.aif – choir like

2. The importance of this topic.

People who share my interests and aesthetic preferences will hopefully benefit from the research I intend to do. If I find that my instrument/composition tool is useable enough I will make it open source.

3. Your intended goals in studying it.

My goal is to make more music using the methods I have described above. At the moment I have four ideas of future multichannel compositions. One for a micro tuned recorded piano, one for string players and electronics, one for an installation with video and the last one for voice and electronics.

4. Major scholarly works in which it is discussed in general.

When building my instrument/composition tool I have been inspired by Fernando Lopez Lezcano's texts about his instrument; CATMASTER AND "A VERY FRACTAL CAT", A PIECE AND ITS SOFTWARE (CCRMA) and A Very Fractal Cat OF CATS, PERFORMERS, COMPOSERS AND PROGRAMMERS (CEC), by texts about John Chowning's Stria, for example A reconstruction of Stria by Olivier Baudoin and STRIA, BY JOHN CHOWNING: ANALYSIS OF THE COMPOSITIONAL PROCESS by Matteo Meneghini (both from Computer Music Journal, 2007) and by Jonathan Harvey's "Mortuos Plango, Vivos Voco": A Realization at IRCAM (Computer Music Journal, 1981).

5. The main kinds of sources you expect to use.

All sources available regarding SuperCollider (the documentation, the book, information on the internet and my supervisor's knowledge) and other relevant articles. Other things that will inspire me is the music of for example Akos Rozmann.

6. The methods you intend to employ in your research.

I intend to continue to work approximately as I have already done for many years. Record potentially interesting sound sources, edit my recordings in a sound editing program like Peak and in AudioSculpt. Look for masked traits that can be revealed etc. I also want to improve my instrument/composition tool. I have many new features I want to add to it, and as it grows it needs to be more structured and split into separate modules.

APPENDIX 12

Conversion table between time format “number of seconds” to “minute.second”

000 - 009	->	**00.00** - 00.09
010 - 019	->	00.10 - 00.19
020 - 029	->	00.20 - 00.29
030 - 039	->	00.30 - 00.39
040 - 049	->	00.40 - 00.49
050 - 059	->	00.50 - 00.59
060 - 069	->	**01.00** - 01.09
070 - 079	->	01.10 - 01.19
080 - 089	->	01.20 - 01.29
090 - 099	->	01.30 - 01.39
100 - 109	->	01.40 - 01.49
110 - 119	->	01.50 - 01.59
120 - 129	->	**02.00** - 02.09
130 - 139	->	02.10 - 02.19
140 - 149	->	02.20 - 02.29
150 - 159	->	02.30 - 02.39
160 - 169	->	02.40 - 02.49
170 - 179	->	02.50 - 02.59
180 - 189	->	**03.00** - 03.09
190 - 199	->	03.10 - 03.19
200 - 209	->	03.20 - 03.29
210 - 219	->	03.30 - 03.39
220 - 229	->	03.40 - 03.49
230 - 239	->	03.50 - 03.59
240 - 249	->	**04.00** - 04.09
250 - 259	->	04.10 - 04.19
260 - 269	->	04.20 - 04.29
270 - 279	->	04.30 - 04.39
280 - 289	->	04.40 - 04.49
290 - 299	->	04.50 - 04.59
300 - 309	->	**05.00** - 05.09
310 - 319	->	05.10 - 05.19
320 - 329	->	05.20 - 05.29
330 - 339	->	05.30 - 05.39
340 - 349	->	05.40 - 05.49
350 - 359	->	05.50 - 05.59
360 - 369	->	**06.00** - 06.09
370 - 379	->	06.10 - 06.19
380 - 389	->	06.20 - 06.29
390 - 399	->	06.30 - 06.39
400 - 409	->	06.40 - 06.49
410 - 419	->	06.50 - 06.59
420 - 429	->	**07.00** - 07.09
430 - 439	->	07.10 - 07.19
440 - 449	->	07.20 - 07.29
450 - 459	->	07.30 - 07.39
460 - 469	->	07.40 - 07.49
470 - 479	->	07.50 - 07.59
480 - 489	->	**08.00** - 08.09
490 - 499	->	08.10 - 08.19
500 - 509	->	08.20 - 08.29
510 - 519	->	08.30 - 08.39
520 - 529	->	08.40 - 08.49
530 - 539	->	08.50 - 08.59
540 - 549	->	**09.00** - 09.09
550 - 559	->	09.10 - 09.19
560 - 569	->	09.20 - 09.29
570 - 579	->	09.30 - 09.39
580 - 589	->	09.40 - 09.49
590 - 599	->	09.50 - 09.59
600 - 609	->	**10.00** - 10.09
610 - 619	->	10.10 - 10.19
620 - 629	->	10.20 - 10.29
630 - 639	->	10.30 - 10.39
640 - 649	->	10.40 - 10.49
650 - 659	->	10.50 - 10.59
660 - 669	->	**11.00** - 11.09
670 - 679	->	11.10 - 11.09
680 - 689	->	11.20 - 11.29
690 - 699	->	11.30 - 11.39
700 - 709	->	11.40 - 11.49
710 - 719	->	11.50 - 11.59
720 - 729	->	**12.00** - 12.09