Design and Evaluation of a Multi-touch Mobile Instrument, Implementing Polygonal Waveform Synthesis

Master's Thesis

MSc in Sound and Music Computing

Gergely Csapo

Aalborg University Copenhagen
The Faculty of Engineering and Science
School of Information and Communication Technology
August 2018

AALBORG UNIVERSITY
STUDENT REPORT

Sound and Music Computing
Aalborg University Copenhagen

Title:
Design and Evaluation of a Multi-touch Mobile Instrument, Implementing Polygonal Waveform Synthesis

Theme:
Sonic Interaction Design

Project Period:
Spring Semester 2018

Project Group:
-

Participant(s):
Gergely Csapo

Supervisor(s):
Stefania Serafin

Copies: 1

Page Numbers: 47

Date of Completion:
August 31, 2018

Abstract:

Mobile-based musical instrument are getting increased popularity, and showing musical potential. However, most currently available applications are not taking advantage of the available technology, especially the capabilities of the touch screen. They indicate a lack of creative solutions as well, by mimicking existing instruments, and following the traditional Windows-Icons-Menus-Pointer paradigm and widget based interfaces. The design of digital musical instruments also involves a great extent of idiosyncrasy. This study presents a collection of relevant research with the aim to frame the design of multi-touch mobile instruments. The findings are demonstrated in the design, development and the evaluation of an interactive musical application for tablets. The evaluation highlighted several additional important considerations for gesture design for multi-parametric control, and outlined the future development for the presented application.

Contents

Chapter 1

Introduction

Smartphones and tablets, as tools for musical performance and expression, are not a new concept. They are accessible, ubiquitous and powerful platform, capable of redefining traditional ways of interacting with music[1][2][3][4][5]. The available music applications are growing, and becoming more and more widespread[6][5]. The potential of mobile music applications, are also highlighted by the emergence of recent tools, designed to support the integration of apps to existing setups, and to render them compatible with other applications and instruments[7][8][9].

Regardless in the increasing capability, and the growing number of musical applications, most designs are not taking advantage of the available sensor technology and the touch screen. Most apps on the market are still following the WIMP(Windows-Icons-Menus-Pointer) paradigm, and featuring, widget based interfaces, which are proved to be unsuitable for multi-touch control, and expressive musical applications. Applications, are also have a tendency to mimic existing instruments, or electronic music interfaces, instead of aiming for novelty[1][3][4][10]. The design of DMIs(Digital Musical Instruments), including mobile instruments, is surrounded by a great extent of idiosyncrasy, highlighting the need for a framework, and the establishment of a design space[11][12][13][14].

The fundament of the investigation presented in this thesis, is motivated by the poorly exploited capabilities of mobile devices, the unsuitability of WIMP and widget based interfaces, and the need for a collection of design criteria around the design of mobile based DMIs.

A recent algorithm for synthesising complex waveforms[15], promotes the design of a new musical interface, and also provides a multiple parameters to interface for multi-touch control. The polygonal waveform synthesis is heavily based on mathematical, graphical representations, creating a solid basis for a novel GUI(Graphical User Interface) design. It has been combined with state-of-the-art anti-aliasing method[16], rendering it a powerful synthesis method, with potential musical application.

It is hypothesised, that more complex interactions can lead to more refined control of multi-parametric interfaces, and that the right design of multi-touch gestures, can enhance the usability of mobile instruments, and promote new directions as well. Unconventional, symbolic GUIs have been implemented in successful applications before[17][18][19][20], suggesting further investigation. The goal of this study, is to present the highlights from the research from relevant fields, and demonstrate them in design and development of a mobile instrument, implementing polygonal waveform synthesis. The evaluation of the prototype is aimed to investigate suitability of multi-touch gestures for parameter control, and use of a symbolic GUI. The study is believed to support future gesture design for multi-parametric control, and mobile instrument design. The presented prototype also initiates an iterative design process, for a multi-touch mobile instrument.

Chapter 2

Graphical Sound

The polygonal waveform synthesis can be traced back to a long history of experimental musical instruments, which were seeking connection between geometry and sound. The term Graphical Sound is related to Russian painter's, Mikhail Tsekhanovsky's name, who attempted to sonify ancient Greek and Egyptian ornaments with the sound-on-film technique(photoelectric process), back in 1929. In 1930 Arseny Avraamov was the first to present experimental musical pieces, based on animated hand drawn images, illustrating geometric shapes and patterns(Figure 2.1). A year later Nikolai Voinov developed a technique, which allowed him to synthesize sound based on paper cutouts with his invention, the Nivotone. Another notable invention in the field of graphical sound was Evgeny Sholpo's optical synthesizer, the Variophone. It was capable of producing entire soundtracks photographed onto film, allowing polyphony via additive synthesis, and it was used for a long time to compose film music before the end of World War II[21][22]. Simultaneously, in Britain and Germany, researchers like E. A. Humphries, R. Pfenninger and O. Fischinger, were developing very similar methods for early graphical sound synthesis[21][23].

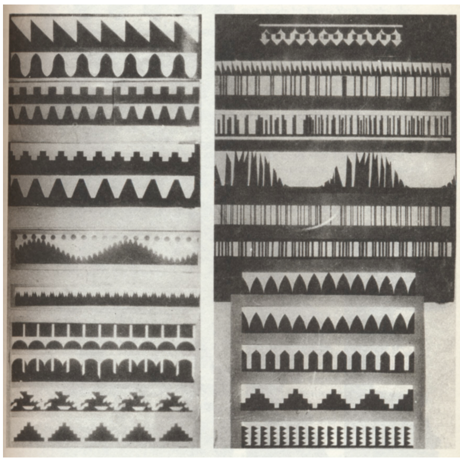

Figure 2.1: Ornaments drawn by Arseny Avraamov[21].

A more sophisticated and recent work, related to the British composer Daphne Oram's name, was the Oramics Machine[24][25]. Her idea was sparkled from inverting the process of a cathode ray oscilloscope, to create an instrument that is able to turn visual information into sound. The Oramics Machine allowed its user to paint on five separate film strips, which were continuously read, and changed different parameters assigned to each strip(pitch, volume, timbre, tremolo, vibrato)(Figure 2.2). Since it was also able to play discrete pitches arranged in time, it resembles how a modern sequencer works.

Figure 2.2: Daphne Oram with the Oramics Machine[26].

Early technologies have layed down the fundament for more recent alternative graphics based sound production methods, such as *wave terrain synthesis*, *scanned synthesis*.[27][28]. A more recent work from Chapman et al., presents alternative methods for deriving waveforms from polygonal structures[29]. The described method of N-gon waves, is primarily based on graphical and mathematical representations of polygons, and motivated by experimental, musical applications. It offers flexible parametrisation of interactive purposes, and strong perceptual connection between the visual representation of the algorithm, and the timbre[29]. This technique shows strong resemblance to polygonal waveform synthesis, which is discussed in detail in Chapter 6.

Chapter 3

Design of Digital Musical Instruments

Modern digital technologies are redefining the idea of musical instruments. Computer-based musical instruments or, DMIs are providing real-time, interactive control for digital sound synthesis, by mapping physical gestures to synthesis parameters. The domain of DMI design offers vast possibilities in creating novel tools for musical expression, and they have been actively investigated by a large community of researchers, such as NIME[30]. This chapter aims to highlight important considerations and practices for designing a DMI as part of this thesis, and investigate mobile-based musical instruments, and the interaction design of musical controllers, and touch surfaces in particular.

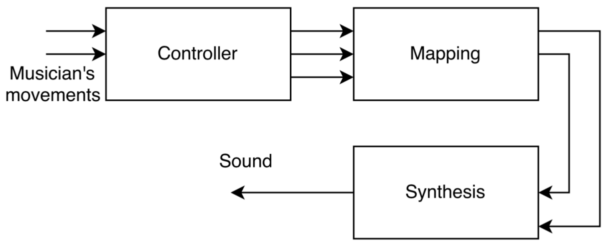

Figure 3.1: Model of DMI's[31].

One way to look at DMIs, is as hardware-software systems, where the raw data is captured from the musicians movements by sensors, then communicated to a computer, which translates them to meaningful parameters, by mapping them to sound synthesis. This three stage model of, controller-mapping-synthesis is a fundamental framework for DMI design(Figure 3.1). A more refined way to approach the design of DMIs, is introduced by the MINUET framework[32], which proposes a detailed model to facilitate both the design and evaluation, by grouping design elements from the player's point of view. The model consists of two major parts, *Goal*, and *Specifications*, describing the the objectives and the necessary tools and considerations to fulfil those. The *Goal* is determined in terms of user experience and context, while the *Specifications* are analysing the potential solutions. The design of a new musical instrument should consider the player and the audience equally, the nature of the activities involved in the performance and the context as well. This facilitates the definition of success criteria which is a crucial aspect of both design and the evaluation process[32]. Some relevant aspects of this framework is discussed in context in Chapter 6.

Visuals could serve as a prominent modality for feedback, especially if no haptics are available, such in the case of touch-screens. Feedback or lack of feedback is necessary part of the human-computer dialogue, which will determine what future actions will be carried out by the player[33]. The CTO of Smule and designer of the successful mobile instrument, the Ocarina[34], Wang, layed down a set of principles for visual design for computer music, from an experience based, artistic perspective[35]. Touch-screens certainly promote a unique angle on visual design, since the surface for interaction and the display overlap[35]. Therefor, visual feedback coupled with physical interaction is suggested to compensate the lack of tactile feedback and to build a stronger perceptual relationship between the actions and the sonic events[35][1]. Regarding general aesthetics, organic, dynamic, but simplistic designs is suggested beyond the functional, which yields a strong connection to the character of the sound[35].

One way to approach the analysis and design of DMIs, is according to the *frameworks and affordances* model. A framework in this context in a physical or conceptual system, that contributes to the performance or composition, which has constraints and affordances. By examining an instruments constraints and affordances, we could understand how those are influencing the music produced by the given instruments, and how they related to instrument specific musical features[36][31]. Affordance in terms of HCI, is defined as a system's perceived capacity for different actions. However, learning a new instrument and uncovering its expressive potential, mainly lies in exploring its constraints and limitations, rather than engaging with its affordances. This is especially true for the freeform nature of digital designs. Physical objects often provide easily identifiable constraints and affordances, although these are harder to point out by initial impressions in complex DMIs, promoting experimentation. Constraint are also addressed as fundamental sources for creativity, mapping out the structure of possibilities, contributing to the construction of mental models. In DMIs, such objective constraints are mainly defined by the hardware, and software algorithm, and partially by the subjective constraints of the musical culture surrounding the instrument[37].

In his two part research, Jorda, emphasises on the players perspective in DMI design[38][39]. He suggests that designers must consider, how a new instrument will impose new ways of thinking, interacting and making music. This is approached from six different angles: *balance; playability, progression and learnability; learning curve; diversity; freedom; and control*[38][39]. An instrument yields balance between challenge, boredom, and frustration, when a sweet spot in complexity is met, where the designs provides depth but does not intimidate novices. New instruments should also be designed with the possible learning curve in mind, appealing to different users accordingly. The learning curve refers to the amount of time needed to be invested to gain expertise, and provide a rewarding experience for the player(Figure 3.2). The learning curve often introduces a trade-off between depth and and ease of learning. The efficiency labelled on Figure 3.2, can be estimated as a ratio between the input and output complexity of an instrument, whereas, the output complexity refers to the sonic diversity, and the input complexity incorporates the DOF(degrees of freedom) of control, and a complexity of the mapping scheme. Ideally, an instrument should be efficient, but still offer depth for musical expression and gradual progression[38]. In case of touch surfaces the DOF equals to the number of touch points times two. Less or equal number of output DOFs than input DOFs, is more suitable for precise parameter control, while more output DOFs are better for complex task, and preferred in music performance[40].

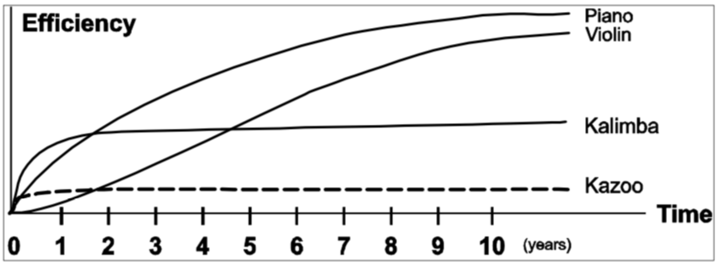

Figure 3.2: Approximated learning of various instruments[38].

Another important angle on new designs, is the sonic diversity it offers for the player, which is primarily rooted in the instrument's constraints and affordances. It can be categorised on three different levels. On the highest level we are interested how well it performs in different context and music styles. Instruments which are more generic tend be be more rewarding and appealing for novices, while, specialised ones are more likely to support individual styles. As for mid-diversity, we consider on how differently an instrument can perform an composition. The lowest level, is aimed at encapsulating how performances of the same composition can differ from each other. This is the closest to the personal level of the player, and relates musical expression[39]. The author also suggests, that an instrument should give a certain level of freedom, by not imposing a certain type of music onto its player, and allowing it to be misused and self interpreted, permitting improvisation[38][39].

Regardless, the various principles and approaches to frame the DMI design space, there is a great extent of idiosyncrasy in play, when it comes to designing new musical instruments[11][12][13][14]. According to Cook, both the design and the music meant to be played with it, is influenced by numerous personal factors, such as the preference of music, knowledge about existing instruments, and the available technology[12]. Although, this aligns with the nature of creative activities, such as composing or performing music, where there is always a certain level of human influence in play.

3.1 Music Interaction Design

As discussed earlier the model of DMIs can be segmented into three stages, the first stage being the interaction with the interface, which is essentially the link between the user and the computer (Figure3.1). Musical interaction design is a branch of traditional HCI(Human-Computer Interaction), focusing on capturing physical gestures, and translating them to musical parameters.

A central goal for HCI designers, is to be able to help users to create an accurate mental model on how a system works, and how it responds to specific actions. One way to facilitate this, is by utilising real-world metaphors[31]. This is related to the theory, which suggests that our conceptual models are grounded in prior experiences. In the context of interaction, it is based on past sensory-motor experiences with physical and digital objects(e.g. flipping a page, turning a knob, using a pinch gesture on a touch screen). The perception of these bodily experiences of motion, space or forces are referred to as *image schemas*. The application of *image schemas* to otherwise abstract metaphors, could facilitate the construction of mental models, and therefore make the interaction more intuitive[41]. Intuitive as, the users is able to subconsciously apply knowledge rooted in prior experiences, *image schemas*[41][1].

Mapping in DMIs, is a separable stage, allowing the explicit definition of the relationship between control parameters and synthesis parameters(Figure 3.1). Regarding the connection between parameters, mappings can be considered as *one-to-one*, *one-to-many*(divergent) or *many-to-one*(convergent). This also corresponds the previously discussed input and output DOFs. It has been suggested, that direct mappings are preferred by novices, and more suitable for precise parameter control, while more complex mappings are favoured for expressive musical control, and gestures[42][43]. In the case of *alternate controllers*, the mapping possibilities are vast, and highly influential on the expressive capabilities of the instrument. To refine the mapping scheme, and improve the flexibility, a multi-layered implementation is suggested, which introduces one or more independent, intermediate layers between input parameters and synthesis parameters. This facilitates the re-mapping of parameters, and a design of more sophisticated parameter relationships[44].

Hunt et al. defined a so called *performance mode* to describe real-time musical systems, where the user is totally in charge of multiple event simultaneously, opposed to an analytical human-computer conversation, where task are performed consecutively, and guided by the system. This well describes the nature of performance oriented DMIs[43]. According to this definition, a system as such, must allow the continuous exploration constraints and affordances, and permit the control of multiple parameters at the same time. The authors also emphasise, that the explorative capacity should extend to editing and configuration operations as well[43]. The exploration of new possibilities, is believed to bear central importance in creative activities.

Instead of the previously discussed, direct and continuous interaction with musical parameters, some DMIs are designed to share the authority on music production process with the player. Musicians interactions with DMIs, can be categorized into three different modes according to that. The *instrumental mode* allows the detailed control of all low level parameters. In the so called *ornamental mode*, the musician gives away some authority, and lets the instrument take over the control over certain processes under the player's supervision. In the *conversational mode*, the player shares the most authority, engaging in an interactive dialog with the computer[45]. Instruments leaning towards the latter two modes, often interface parameters for higher level processes, opposed to note-level intricacies. This category often involves composition oriented applications. When the player has to supervise and control several processes at the same time, visual feedback has a significant impact on the human-computer dialogue[18]. The term *interactive composing*, can be connected to John Chadabe's name, who defined it as the real-time interaction with a computer music system, where composition and performance takes place simultaneously[46]. This is rooted in his early works with programmable sequencers. In his definition he also implies a certain level of unpredictability and shared authority with the system, creating a mutually interactive relationship[46]. Interactive composing, and *conversational* interaction mode, is more suitable for fine parameter control, which is the subject of this study.

3.2 Multi-touch surfaces and Tabletop Displays

Multi-touch interaction, is not just commercially successful, but due to its popularity, gestures like swipes and two finger pinch-zoom, became natural for interacting with touch surfaces[47]. Multi-touch gestures are also considered natural, and capable of supporting creative exploration, due to their multi-dimensional nature[1]. Touch surfaces are convenient, non-intrusive tools for capturing fine gestures, produced by the fingers, rendering smartphones and tablets perfect candidates for input devices for DMIs. This section will further investigate the potential of touch surfaces as DMI controllers, with the aim to support the interaction design of mobile instruments.

Tabletop applications of multi-touch surfaces share characteristics with larger handheld interfaces such as a tablet, in their nature of interactions, therefore, studies considering interaction with tabletop displays, are also subject of this investigation[33][48][40][49][50][51][52].

Gonzalez-Inostroza et al. is proposing a new model for approaching interaction design for multi-touch surfaces[31], by dividing the *controller* stage into two additional stages, *interaction framework* and *processing framework*(Figure 3.3). This division aims to answer how the user interacts with the touch surface, and ultimately how it translates to control parameters. In other words, we can consider the first stage as raw touch data, and the second one as interpreted action. For further classification of the *interaction framework*, the authors distinguish between, keys, sliders, and multi-dimensional zones, as they are different in terms of what actions they promote, and what information they can capture. Regarding the *processing framework*, it is possible to differentiate between the following ways of analysing touch data:

Vertex: Analysis of the position of individual touch points, and their relative angles or distances to each other.

Polygon: Analysis of interrelated touch points, as an estimated posture, defined by relative angles, positions and number of points.

Gesture: Analysis of touch data over time, which could provide information about movement and gestures.

The concept of *interaction framework* and *processing framework* offers a new angle on how users could construct metal models about new system, and how designers could promote certain concepts to facilitate that[31]. As the concept of *processing framework* also implies, that the relationship between individual points could provide rich information about the users' action, and lead to new ways of designing and interpreting touch gestures. The relationship analysis between touch points is able to extend the number DOFs, increasing the available complexity[53].

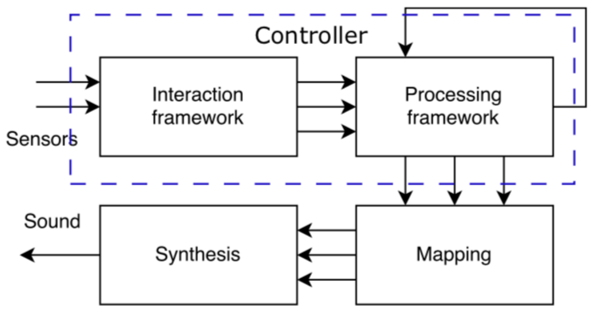

Figure 3.3: Interaction framework and processing framework division of the standard controller stage[31].

HCI researchers are trying to expand the popular gesture set, that people are accustomed to(e.g. pinch, swipe, double-tap, etc.)[54]. Numerous researches have been conducted in evaluating new control schemes and gestures, designed for multi-touch interactions[33][48][40][50][51][52][54][55].

A notable design is the *anchor point* interaction, in which case the first contact point is treated as an anchor point for subsequent points, enabling the calculation of the distance, and the angle between two points(Figure 3.4). A study, comparing the mapping of *anchor point* interaction to the mapping of global screen space coordinates, concluded that the prior supports exploration and creative expression better, even though it is proved to be less intuitive[55].

It has been proved, that humans perceive rotation and translation as inseparable actions, suggesting a compound interaction method for such tasks[52][40]. Although, precision and coordination are conflicting design goals, in terms of output DOFs required, and subjective factors such as fluidity and learnability also influencing user preference[40][48].

Results from a study conducted on users' preference over various touch gestures, indicate that, the participant generally preferred simpler gestures. One hand over two hands, one finger over multiple fingers. The preference also leaned toward conceptually simpler gestures, ones which are rooted in real world metaphors, and analogies[48].

When designing interactions for multi-touch surfaces, it is important to consider how hand position will change during the course of actions. An investigation in action planning for multi-touch gestures, showed that for rotation tasks, users tend to adjust their hand position prior to the action. This highlights, comfort, as an aspect to take into account for designing gesture interactions[54][49].

The action planning or *prospective movement control* is mainly influenced by constraints and limitations of hand postures, as users prefer actions, which result in an optimal *end-state comfort*. In other words, in order to avoid stress or discomfort, users will try to adjust their hand accordingly. The initial grasp of a physical or

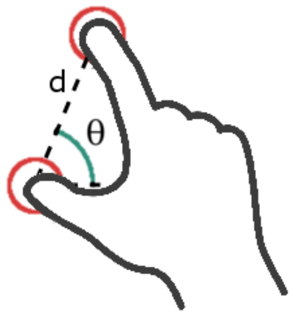

Figure 3.4: Anchor point interaction, with distance d and angle θ.

virtual object significantly depends on the object's shape and size, and the course of the actions. For rotation tasks, anti-clockwise movement is preferred[49].

In the attempt to develop design principles for multi-touch interaction, Wu et al. divided gestures to *registration* and *relaxation* phases[51]. The *registration* phase refers to the beginning of the interaction, when the gesture recognition takes place. By setting the context with a distinctive posture, the gesture move to the *relaxation* phases, where subsequent dynamic actions can happen, not restricted by the initial rules of recognition. This resolves the tension posed by the initial stage of the interaction, allowing the user to comfortably perform continuous, freehand actions, and control the mapped parameters. The interaction design based on this two stage concept, could also benefit from the re-usage and combination of simple gesture. It would ease the cognitive load, imposed by memorising a large set of gesture, and would allow the construction of compound gestures, with the same intuitive *relaxation* phases[51]. Gestures which only consist of *registration* phase, are considered as discrete gestures(e.g. double tap)[33].

The design aspects discussed in this chapter are revisited and summarised in the context of designing an mobile instrument in Chapter 6.

Chapter 4

Mobile Instruments

Smartphones are getting more and more widespread, and this tendency is reflected in the number of applications being released in various fields, including music[6][5][1]. Mobile devices have reached a the computational capability to be considered as platforms for developing DMIs, giving way for the emergence of mobile instruments[2][3][4][5].

Designing new instruments for mobile devices, require the examination of constraints and affordances rooted in the platform itself, and it is important to understand the surrounding culture and user experience as well. As part of a growing research field of mobile instruments, Tanaka et al. carried out a survey, investigating commercial software with the aim of highlighting themes around GUIs and interaction. According to this survey, most applications feature some sort of sequencing and parameter tweaking as primary mode of interaction, while *instrument-like* and *instrument-inspired* apps considered as *"toys"*. This implies that imitations of real instruments do not live up to the potential of mobile devices, and promotes a different approach in designing mobile instruments. The limitations of the screen size and GUI capabilities, points toward gestural interaction and adaptive interfaces proposed by the authors[3].

"I would prefer to use something that treats the device as an interface it its own right and uses gestures/controls more natural to it than trying to play a tiny piano."[3].

The WIMP paradigm, which is universal on desktop interfaces, successfully standardised the way GUI elements are designed, and accessed, although, it also brought limitations to the interaction possibilities, and stuck to two-dimensional concepts. As touch-screens on mobile devices become more and more common, the necessary reinterpretation of the WIMP paradigm became clear. Currently available touch-screens allow the tracking of multiple fingers, and as the available screen space of mobile devices are significantly smaller then on desktop computers, issues such as occlusion appeared, promoting new, creative solutions in GUI design for touch surfaces[56]. However, WIMP style, and widget based interfaces are still the most common in musical applications as well, where the widgets usually take the form of hardware controllers, such as knobs, slider, buttons(skeuomorphic interface elements). Relying on such embedded cultural knowledge can facilitate the construction of mental models, but certainly does not align with musical performance in mind, and it is not able to take advantage of multi-touch interactions[10][1]. The skeuomorphic, widget-based design is better suited for editing and configuring task, such as compositional apps require, however, multi-touch gestures are more appropriate for performance and the exploration of potential musical material[1].

To provide a better insight into the design space of mobile instruments and music apps, Kell et al. presented two consecutive studies reviewing mappings in existing iOS applications[4][5]. Majority of the mappings are based on metaphors, and show resemblance to existing instruments or electronic musical interfaces[4][10]. Metaphors are refer to commonly known concept or system, in order to help understanding the rules and behaviour of a new one[31]. Research suggests, that the familiarity of virtual objects to physical objects, can improve the interactions, by leveraging prior knowledge to help users predict the the interface's behaviour[54]. The pitch is mostly assigned to a piano-like interface(mapped from left to right discretely) or mapped from bottom to top. Timbre often changed according to presets, like in digital drum pads(each pad represents different part of the drum kit). Time progression traditionally goes from left to right, often in a discrete, step sequencer fashion. Volume is continuously mapped to from bottom to top in most cases. The triggering of sounds are either done by a sequencer or via the touch screen, one sonic event per touch point[5].

Overall, the lack of creative mappings and the insufficiency of the WIMP paradigm suggests, that the design of mobile instruments are not thoroughly explored yet, and it is believed, that touch-based musical applications will increasingly continue to receive attention from a wide community of users[4][5]. This sets the stage for innovation in mobile instrument design, and future investigating in the field.

Chapter 5

Related works

This chapter will discuss several related works, selected based on their relevance to this project, covering various different design aspects.

ReacTable is a tangible, tabletop, modular, compositional interface. The sound production is created from the interplay of various signal processing modules(e.g. oscillators, filters, envelopes), which are symbolically represented with projected visuals and physical tabletop objects, avoiding numerical or textual information[18]. *ReacTable* has gone through various iterations since its first prototype, and the creators recently released a version developed for mobile platforms. The mobile version is using multi-touch interaction to manipulate GUI elements instead of tangible ones(Figure 5.1). The full digital implementation have came with flexibility and distributability, but sacrificed the benefits of the haptic feedback and the available screen space[57].

Peschke et al. developed an oscillator algorithm, based on sampling of arbitrary periodic shapes, defined by bezier curves[58]. The bezier path is periodically scanned at the specified frequency to sample the unit normal vector at each point, and fill a wavetable. The parametrised curves can be controlled via handles assigned to control points(Figure 5.2). The shape of the geometry shows strong perceptual relationship to the oscillator's timbre, supporting the graphical representation of the algorithm for the user. The authors suggest less computationally expensive, parametrised geometries, such as polygons[58]. The peculiar set of software instrument, *DIN*, is also using bezier curves to allow users to construct, periodic waveforms. Visual representation of sound plays an important role in its GUI design[59].

Figure 5.1: *Reactable* mobile app interface[57].

The iPad app, *TC-11* takes advantage of the relationship analysis between individual touch points, and temporal information, creating a rich, gestural interface for music performance. The GUI of the app is free from widgets and WIMP elements, putting emphasis on visual feedback on sound and interactions alike. *TC-11* also allows to customisation of the mappings, rendering it a highly sophisticated and flexible mobile instrument[60].

Previous work from the author, is a smartphone application, designed for recording and playback of short sound snippets, utilising touch gestures[61]. The application allows the real-time manipulation of various synthesis parameters, by extracting temporal, and spatial information from touch data. Evaluation of the app, supports the use of multi-touch interactions for controlling sound parameters[62].

Borderlands, is granular synthesizer developed for iOS, which relies heavily on multi-touch controls and audio-visual synergy[63]. The design is motivated by removing traditional musical paradigms related to the underlying synthesis technique, and instead, provide an interactive visual representation for musical parameters and processes. The aesthetic simplicity and playfulness of the interface, is intended to bypass the learning curve, and promote exploration and experimentation in the musical design space[17].

Similarly to Borderlands, *FMOL*, a web based DMI, is also designed with strong visual feedback in mind. Nearly all aspects of of the synthesis process is visualised in a symbolic, non-technical way. Many GUI elements serve both as representation

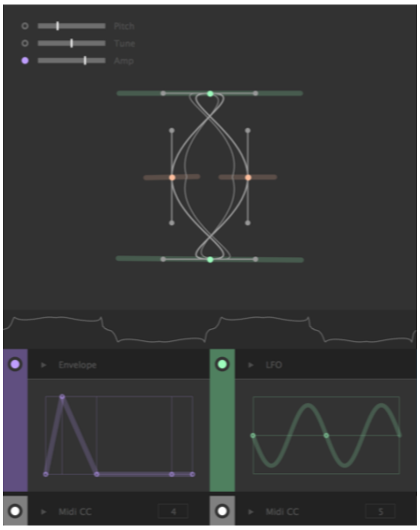

Figure 5.2: Interface of *Cyclone*(**top**: geometry with control handles, **middle**: calculated, **bottom**: additional waveshaping modules)[58].

and control for the system, creating a strong perceptual connection between the sounds and the interactions. The detailed visual feedback facilitates the simultaneous control of multiple parameters at the same time, which would be otherwise hard to monitor relying only on the auditory channel[19][18].

Figure 5.3: Interface of Borderlands[63].

NodeBeat is popular application, among the examples on alternative compositional apps. It features a primarily symbolic interface, where different timbres are assigned to different colours, and the triggering of sequenced notes is based on relative spatial relationships, opposed to traditional left to right timeline[20].

Chapter 6

Application

6.1 Concept

The application proposed in this chapter is motivated by the previously discussed research(Chapter 3, and 4) and related works, with the aim to demonstrate the findings around the concept of multi-touch mobile instruments, and to provide an end-user implementation for the study on polygonal waveform synthesis[15].

The prototype is designed along the lines of an alternate controller, supporting conversational and instrumental interactions as well, with interactive composing and experimentation in mind. It is believed that this provides the freedom and flexibility for both exploration and observation of the underlying algorithm. The application is designed to yield sequencing capabilities, and real-time, low-level parameter control. The interfaced parameters are derived from the core algorithm of polygonal waveform synthesis, and the added envelope and sequencer features. Classic oscillator applications, only provide control for amplitude, frequency and the timbre by the selection between a few preset waveforms(e.g. sine, sawtooth, square, etc.). The application is designed for larger screens, such as the iPads, and tabletop hands free use.

6.1.1 Mapping

The mapping mainly follows direct, one-to-one relationships, to allow fine and accurate control. Although, some parameters pairs are coupled in a single compound interaction, following divergent relationships. The interface also permits the control of multiple oscillators at a time, therefore, to interact multiple parameters simultaneously, adding to the dimensions of control. The relationship analysis of touch points provides an intermediate mapping layer, which enables the flexible customisation of the mapping scheme. The base set of parameters derived

Table 6.1: Mapping between touch points, intermediate parameters and synthesis parameters(f_0 - fundamental frequency, r - radius, n - order, T - teeth, Φ - phase offset)

Processing framework	Gesture alias	Touch data	Intermediate parameter	Synthesis parameter
Gesture	Trajectory	(x_1, y_1)	direction regression	f_0
Vertex	Anchor	$(x_1, y_1), (x_2, y_2)$	distance	r
			angle	n
Polygon	Triangle	$(x_1, y_1), (x_2, y_2), (x_3, y_3)$	area of triangle	T
			mean angle	Φ

from the algorithm of polygonal waveform synthesis, are controlled, via three different interactions, corresponding to the three different proposed ways to analyse touch data, according to the processing framework(Table 6.1)[31]. The mapping scheme and a simplicity of the interface, comes with low macro diversity and a low learning curve, supporting a specific music style rooted in the nature of the interface[39].

6.1.2 Gestures and Interaction

The three main gestures, designed to control synthesis parameters, are the following:

- **Trajectory:** output - the direction of circular regression of a single point's trajectory, registration - tap and hold with one finger
- **Anchor:** output - distance and angle between two points, registration - interact with two fingers
- **Triangle:** output - the mean angle to the centroid of three points and the area defined by a triangle, registration - interact with three fingers

All designed with traditional knobs in mind, as they all require grasping(registration) and rotary motions. This is expected to be supported by users' prior experience with electronic musical interfaces, and therefore improve intuitiveness. The knob like behaviour is taking advantage of the digital domain, by allowing to register the gesture anywhere in the proximity of the object, and the relative manipulation from any initial grasping position, to improve the comfort of the interaction. The relaxation phase of the gesture facilitated by allowing the user to perform the rotary action anywhere on the screen, once the gestures is registered. The three proposed comes with different level of complexity, as each touch point adds two DOFs for the input. Therefore, the *trajectory* gesture has two input DOFs and one

Table 6.2: Additional gestures and their mapped functionality

Interaction	Function
double tap	create oscillator
tap and move	drag object
tap and hold	enter "idle mode"
drag to bottom of the screen	delete oscillator
position along X axis	set position in loop
position along Y axis	set envelope duration

output, the *anchor* has four inputs, two outputs, and the *triangle* has six inputs and two outputs. This difference in complexity, is expected to show different in users preference between the gestures.

All the gestures can be carried out with one or two hands as well. Table 6.2 list the additional gestures covering the rest of the applications functionality.

6.1.3 GUI and Aesthetics

The interface intended to reflect novelty, simplicity and playfulness. The GUI is strictly symbolic(shapes, colours, animation), free from widgets and textual or numerical information. All GUI elements are bi-directional, in terms of providing feedback and enabling interaction, inspired by the success of related works discussed in Chapter 5. The aesthetics close representations of the underlying algorithm, creating strong perceptual relationship with the synthesis, the gestures and various sound parameters. According to Wang, visualising an algorithm can help understanding it, and the coupling between visuals and audio facilitates interactions[35]. The built in sequencer triggers the oscillators on a continuous timeline, spreading from left to right, just like in conventional applications. The oscillators' transparency is animated according to the envelope's output to provide visual feedback on timing, and the saturation is mapped to the fundamental frequency.

6.2 Implementation

6.2.1 Platform

The application was developed, using the JUCE[64] framework, and distributed through the Appstore[65], under the name, *Posc*. Mobile instruments and touch-music apps are predominantly developed for iOS, and showing increased popularity among both users and developers[4][5]. Furthermore, the operating system was picked due to performance considerations. Distribution and public exposure is intended the collect feedback from remote users for future development.

6.2.2 Synthesis

The polygonal waveform synthesis is based on the sampling the points of a polygon by a phasor with angular velocity based on the fundamental frequency, and calculating their projection(Figure 6.1). The radial amplitude p of the polygon can be calculated as it follows:

$$p(\phi, n, T, r) = \frac{cos(\frac{\pi}{n})}{cos[\frac{2\pi}{n} \cdot mod(\frac{(\phi+\Phi)n}{2\pi}, 1) - \frac{\pi}{n} + T]} \cdot r \tag{6.1}$$

where $\phi(t)$ is the phase angle at time t, n is the order(number of vertices), T is the *teeth* parameter, and r is the radius. The order of polygon can be defined by its schlälfli symbol n=(p/q), allowing the generation of regular star polygons by irreducible fractions(Figure 6.2). With star polygons p defines the number of vertices, and q defines the number of periods needed to describe a closed shape.

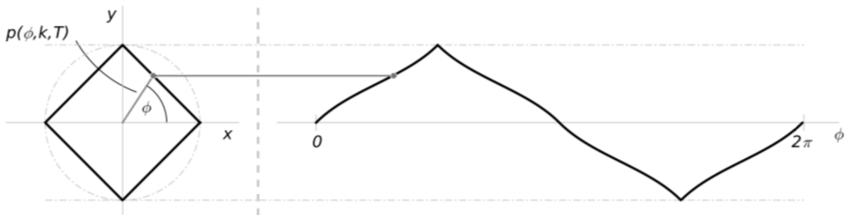

Figure 6.1: Polygon, and its sampled projection in the time domain(n=4)[15]

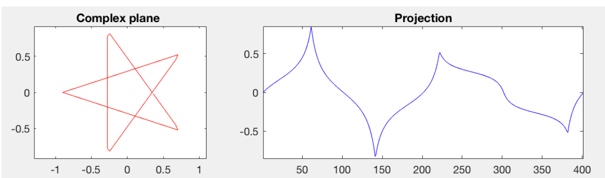

Figure 6.2: Pentagram, and the first period of its respective waveform(n=2.5, Φ=$\pi/5$)

The discrete phase increment per sample is calculated by:

$$\theta = 2\pi f_0(\frac{1}{f_s}) \tag{6.2}$$

with fundamental frequency f_0, and sampling rate f_s. The discrete time-domain projection to the y axis is acquired by the following equation, where n is the sample index:

$$y(n) = \Im\{p(n) \cdot (cos(\theta) + jsin(\theta))\} \tag{6.3}$$

This altogether, provides six parameters to interface, although the current implementation of the prototype, only uses convex polygons therefore, integer schläfli is used, as order n, resulting in five core synthesis parameters:

- n - order
- f_0 - fundamental frequency
- Φ - phase offset(initial phase angle)
- θ - phase angle
- T - *teeth*

The polygon order n defines the harmonic intervals in the generated waveform, the higher the order, the bigger the distance in frequency between two harmonics. With the increase in order, the polygon eventually turns into a circle, producing a sine wave, having only the fundamental present. The *teeth* parameter applies, visual distortion on the polygon, resulting in added partials in the signal[15]. The increase in *teeth* would also result in clipping, as the radial amplitude exceeds the unit circle, to prevent this, the application implements adaptive normalisation.

6.2.3 Anti-Aliasing

Due to the nature of the digital oscillator algorithm, aliasing is inherently present. Aliasing can be traced back to the discontinuities(jumps) in the signals first derivative, which can be analytically located in the waveform of polygonal oscillators, suggesting the application of the polyBLAMP method[16]. An alternative to this would be to over sample the signal, filter it with a high order FIR low-pass filter, with the cutoff at Nyquist frequency(half of the sampling rate), and down sample it back to original[15][66]. Although, the prior proved to yield better results, by better retaining the amplitude of higher harmonics, and being less computationally costly[16][66][67].

Table 6.3: Coefficients for the four-point polyBLAMP residual[16]

[-2T , -T]	$d^5/120$
[-T,0]	[-3$d^5 + 5\text{d}^4 + 10\text{d}^3 + 10\text{d}^2 + 5\text{d}+1]/120$
[0,T]	[3$d^5 - 10\text{d}^4 + 40\text{d}^2 - 60\text{d}+28]/120$
[T,2T]	[-$d^5 + 5\text{d}^4 - 10\text{d}^3 + 10\text{d}^2 - 5\text{d}+1]/120$

According to the BLAMP(Band-limited Ramp) method, the samples of a trivial waveform are corrected, by positioning the scaled BLAMP residual function at the location of discontinuities, and adding the derived correction coefficients to the neighbouring sample values. The BLAMP residual is acquired as the difference between a trivial ramp function and the second integral of a band limited impulse(Figure 6.3). To improve the computational efficiency, the band-limited impulse is approximated by a four-point polynomial, leading to the polyBLAMP method, used in this implementation[67][66].

To apply the 4-point polyBLAMP, the exact location of each discontinuity, the four neighbouring samples have to be located. Discontinuities occur at every vertex, therefore at $2\pi/n$ phase angle. In discrete time, with added phase offset, every exact kth location can be calculated as it follows:

$$n_{disc} = f_s/(nf_0) \cdot k - f_s/f_0/(\frac{2\pi}{\Phi}) \tag{6.4}$$

and the neighbouring samples n_1, n_2, n_3, n_4 are simply found as the ceiled and floored indices around the discontinuity. The respective correction coefficients are acquired from Table 6.3, where d is the fractional delay between the exact discontinuity and the next sample(n_3):

$$d = ceil(n_{disc}) - n_{disc} \tag{6.5}$$

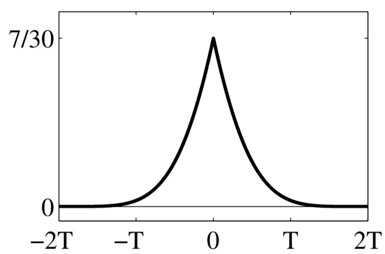

Figure 6.3: Four-point polyBLAMP residual(T - sampling interval)[67]

Before the function is added to the waveform, it is scaled according to the slope of discontinuity(μ). The numerical solution, with linear interpolation can be calculated as:

$$\mu = 2\left|f''[n_2] + (n_{disc} - n_2) * ((f''[n_3] - f''[n_2])/(n_3 - n_2))\right| \tag{6.6}$$

where f'' is the second order difference of the discrete signal. As the last step, the scaled correction coefficient, calculated from the Table 6.3, are added to the respective neighbouring samples with retained sign. Figure 6.4 shows the synthesised signal and its corrected version. This method is not suitable if the *teeth* parameter is used, due to the doubled discontinuities, and the differentiation introduces inaccuracy rooted in the time resolution of the signal. The real-time application, synthesises the waveform at audio-rate, and applies polyBLAMP correction at each frame, but the polygon is only regenerated if the synthesis parameters are changed.

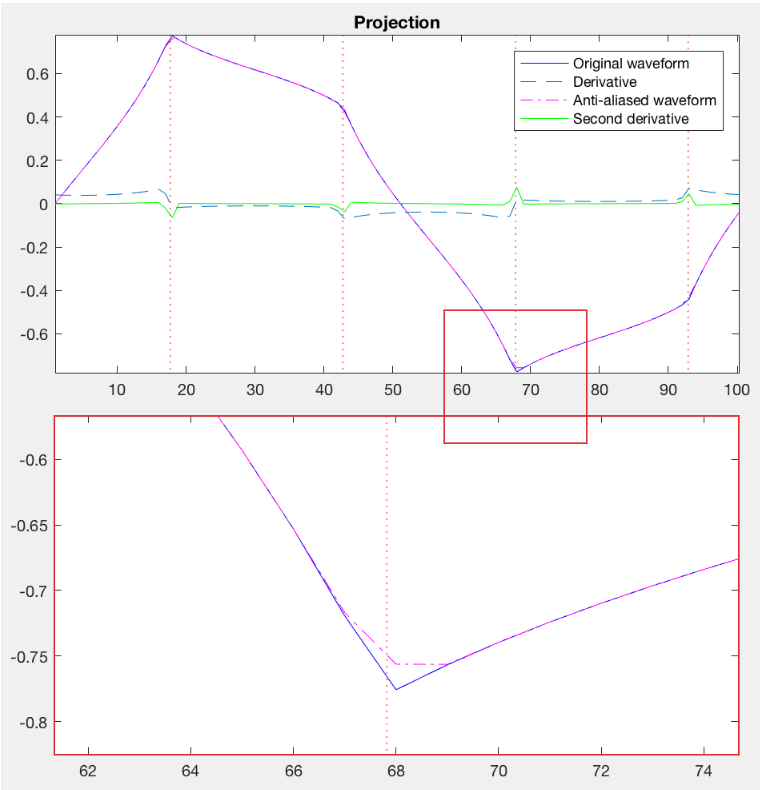

Figure 6.4: Synthesised waveform, anti-aliased waveform, first derivative, second derivative(n=4, Φ=$\pi/6$)(The horizontal dotted lines, marke the exact location of discontinuities)

6.2.4 Software

The application is written in C++, using the JUCE framework. The is available in the digital appendix, along with the class reference documentation. The following subsection aims to give an overview on how each class is implemented and what functionality it covers.

MainComponent, is the main class for handling audio, and holding individual oscillator instances, under *OscInstance* structs. Figure 6.5 shows the class hierarchy for oscillator instances. Each instance has its own *FirLpf*, low-pass filter class, its owns *Sequencer* class, responsible for timing, *OscComponent* class, handling the graphical representation and interactions, and *Envelope* class. Further down the hierarchy, each *OscComponent* has its own *Oscillator* class, responsible for synthesis and anti-aliasing, and *TouchHandler* class, taking care of the touch data processing.

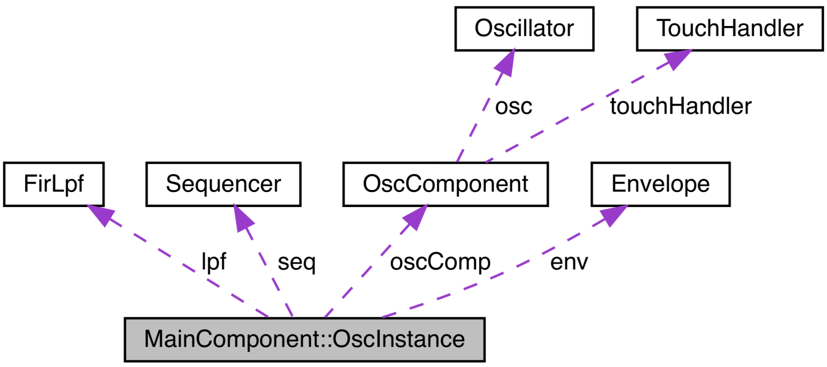

Figure 6.5: OscInstance struct hierarchy

The *OscComponent* class takes care of all the graphics and animations(Figure 6.6). It also handles parameter the registration and mapping of gestures, taking the role of the *interaction framework*. The gesture parameters are not directly linked to the synthesis parameters, but instead, the difference between past and current values is calculated. For more details see the digital appendix.

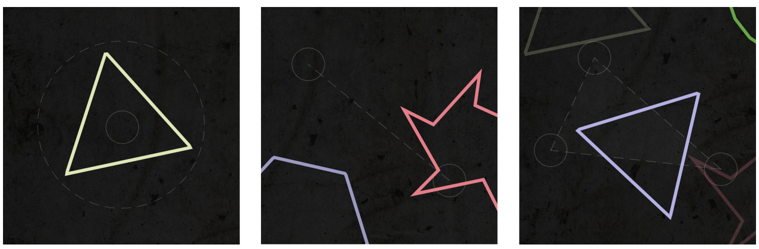

Figure 6.6: The respective indicator for the three main gestures for controlling the synthesis parameters(**left:** trajectory, **middle:** anchor, **right:** triangle)

The *TouchHandler* class is designed to fill the role of the *processing framework*, process raw touch data and analyse relationships between touch points. Its private functions(Figure 6.7) are all implemented to calculate various parameters, solely based on touch points, which are accessible by public *getters*. The *getCircularRegression()* function, checks whether the last sampled point lies on the circle which has been defined by the previous three, with a predefined threshold in radius. If yes, it returns the angle between the last two points.

For further details on the software implementation, please refer to the class references or the source code included in the digital appendix.

Public Member Functions

	TouchHandler ()
	~TouchHandler ()
void	**addTouchPoint** (const MouseEvent &e)
void	**rmTouchPoint** (const MouseEvent &e)
int	**getNumPoints** ()
void	**updatePoints** (const MouseEvent &e)
Point< float >	**getTouchPos** (const int &i)
void	**sampleTouchPointCoordinate** (const MouseEvent &e)
float	**getAnchorRadiusDelta** ()
float	**getAnchorAngleDelta** ()
float	**getTriAreaDelta** ()
float	**getTriRotationDelta** ()
float	**getCircularRegression** ()

Static Public Member Functions

static int	**negMod** (const int &n, const int &m)
static float	**linToLog** (float x, const Range< float > &inRange, const Range< float > &outRange)
static float	**linToExp** (float x, const Range< float > &inRange, const Range< float > &outRange)

Private Types

typedef struct **TouchHandler::TouchPoint** **TouchPoint**

Private Member Functions

float	**getDist** (const Point< float > &a, const Point< float > &b)
float	**getNormalizedDist** (const Point< float > &a, const Point< float > &b)
float	**getAngle** (const Point< float > &a, const Point< float > &b)
float	**getNormalizedAngle** (const Point< float > &a, const Point< float > &b)
float	**getNormalizedArea** (const Point< float > &a, const Point< float > &b, const Point< float > &c)
float	**getNormalizedRotation** (const Point< float > &a, const Point< float > &b, const Point< float > &c)
Point< float >	**getTriCentroid** (const Point< float > &a, const Point< float > &b, const Point< float > &c)
Point< float >	**getCircleCentroid** (const Point< float > &a, const Point< float > &b, const Point< float > &c)
	JUCE_DECLARE_NON_COPYABLE_WITH_LEAK_DETECTOR (**TouchHandler**)

Figure 6.7: Class reference for TouchHandler class

Chapter 7

Evaluation

When it comes to evaluation DMIs, ergonomics, and efficiency is equally important to the subjective, experience based qualities of the instrument. Numerous research have been trying to frame the otherwise mostly idiosyncratic methodology of evaluating DMIs. The following approaches derived from these, serve as building blocks for creating a tailored evaluation method for the presented application.

User centered designed is a well known methodology in HCI, and common in evaluation conventional human-computer interfaces[11][13][11][68]. Gesture design in specific is suggested to follow participatory design methods, which often includes iterative design and the involvement of end users both for design and evaluation[48].

The evaluation process must start with the definition of a set criteria(or the redefinition in iterative design). This improves the both the validity and the replicability of the experiment, and helps contextualising the findings[69].

The term usability, incorporates effectiveness, efficiency and subjective satisfaction regarding a product. The general evaluation of usability is focusing on a given set of a criteria, which are usually considering the following aspects of the design: *learnability*, *effectiveness*, *efficiency*, *satisfaction*, *capability*. These criteria, serve as the basis for the development and evaluation of software applications. This is putting the focus on how well the intentions of the player are translated by the interface[6]. This traditional usability approach is revolving around task based interaction with WIMP interfaces, and it proved less suitable for musical interfaces and systems designed for creative activities[11][13][45]. However, the task based approach, allows the clear investigation of a target subject, whether if its a specific interface element, interaction, or action[11][68]. Wanderley et al. is approaching the evaluation of the usability of musical input devices from the dimensions of *learnability*, *explorability*, *feature controllability*, *timing controllability*, with the utility of simple musical tasks. This is motivated by scenarios, where musical expression and performance plays a central role, which is potentially moer suitable for evaluating DMIs[14][13].

Recent HCI trends promote user-experience studies for evaluating musical interfaces, instead of the classic task based usability approach[11][70][68][45]. This proved especially suitable for novel interfaces, where highly subjective aspects, such as impression or aesthetics play an important role. Study suggest that the selected methodology should to be tailored for the system's specification, based on fundamental nature of the activities related to the system[11][69]. Quantitative methods are suitable for verification, confirmation, or statistical analyses, qualitative methods are useful to gain an understanding on the users' mental models, and other unquantifiable measurements, such as motivation[11]. Due to the benefits and limitations of different tools, mixed evaluation methodology is promoted. Quantitative methods can verify, hypothesis emerged from qualitative investigation, or qualitative methods can provide additional depth to quantitative findings[11]. The triangulation of methods also contributes to the validity of the experiment.

For performance oriented, multi-parametric interfaces, it is believed that the accuracy and capability of control will improve over time. Short experiments therefore, only able to reveal information about initial impressions from an analytical perspective, disregarding a broader user experience insight with the new interface. On the contrary, longitudinal test could highlight more intimate information, rooted in the changing relationship between the user and the instrument over time(e.g. learning curves, explorability, effects of novelty, etc.)[43][68]. The longitudinal approach allows the integration of an instrument into a real-life context, and monitor the evolution of the player-instrument relationship facilitated by the given freedom[68].

7.1 Method

The evaluation of Posc was motivated by acquiring new design requirement for future development, the comparison of different touch-point processing frameworks' suitability for real-time synthesis parameter tweaking, and the assessment of a solely symbolic GUI in a real-time musical application. This is also believed to provide valuable information for gesture design for multi-parametric control. The motivation can be summarised in the two following research questions:
Is vertex, polygon, or gesture based interaction more preferred, than the others for parameter tweaking?
Can a solely symbolic GUI convey satisfactory amount information about polygonal waveform synthesis?

In order to evaluate the general usability of the instrument, a set of success criteria were formulated based on the proposed dimensions(*learnability*, *explorability*, *feature controllability*, *timing controllability*) by Wanderley et al.[13]:

- The participants found the interface easy to use.
- The participants got engaged in free experimentation.
- The participants found the interactions accurate and responsive.
- The participants felt in control of the triggering of sounds.

Both qualitative and quantitative methods were chosen to cover usability and user experience aspects of the evaluation. The primary sources for data gathering, were the think-aloud method and the post test questionnaire, which were triangulated with additional observations noted by the conductor. The think-aloud method was aimed to provide additional qualitative feedback about the users' experiences, and the evolution of their metal models[70]. The questionnaire consisted of four sections. The first considering demographics, investigating familiarity with musical interfaces. The second, lists likert scale items focusing on usability, covering each above listed criteria. The third lists questions about to parameter tweaking with the designed gestures, and the fourth one poses open-ended questions allowing the users to elaborate on their experiences(Appendix A).

7.2 Setting and Procedure

The evaluation has been taken place in a semi controlled environment, with the presence of a conductor. A total number of 14 participants were involved through convenient sampling, between the age of 24-35. The procedure consisted of two parts. The first stage was designed to highlight usability issues and to level out the initial learning curve, where each participant asked to complete a list of tasks, to cover all basic functionality, and familiarise them with the interface. The second part was focusing free experimentation and exploration. In both parts, the users were asked to think-aloud and encouraged to comment their actions and experience. The conductor note prominent comments, along with other observations. The elapsed time was measured in both stages, to provide additional quantitative data.

7.3 Results

The 9 out of 14 participants were active, or occasional users of hardware controllers, and all of them were familiar with mobile music apps.

7.3.1 Usability

The usability have been evaluated by 13 likert scale items covering the four evaluation criteria listed in the section above. Most participants found the application easy to use, regardless the general confusion, which have been reported and observed as well.

The controllability also received positive feedback. Although, several issues were hihglighted by the think-aloud method, the open-ended questions and the observations. One of the common issues, was the gesture registration, when users placed their finger too far from the objects or double tapped too slow. Similarly, the object's size seemed to interfere with the interactions, when users placed their fingers too close together, making them unable to perform their actions. Not knowing the parameter limits also, appeared to interfere with the controllability, and some participants deleted objects accidentally, by dragging them too close to the bottom of the screen. Several answers and comments were highlighting the desire to control pitch in a discrete steps, such as a chromatic scale. Others, lacked the ability to control the tempo. Some participants, noted that they would like to sustain the sound, while manipulating the parameters.

The participants spent an average five minutes for free experimentation, and majority reported, that they felt there is more to explore, and would play it more. The final compositions ranged from random explorations, to thoughtful sequences.

The automated sequencing of the oscillators, and the continuous timeline received mixed feedback. Some users preferred the free, organic sequencing, while others highlighted the need for discrete steps and visual indicators to facilitate rhythmic compositions. Users also pointed out the lack of ability to control the tempo.

7.3.2 Gesture Preference

10 out of 14 particpants found the *anchor* gesture the most intuitive. The comfort, ease of use and accuracy, received indicated preference leaning toward the *trajectory* gesture though. According to the overall gesture preference, *anchor* proved to be the most suitable, *trajectory* the second, and *triangle* the least(Figure 7.1).

Additional feedback from the questionnaire, confirmed that people perceived the *trajectory* gestures more free from constraints and comfortable. The *anchor* gestures was reported to be natural and intuitive, and it related well with the object scaling. The *triangle* gesture, was described as uncomfortable, and less natural.

One participant even argued that, his prior experiences are supporting one and two finger gestures.

Observations highlighted, that some participants were using two hands to perform the gestures, and often utilising unexpected finger combinations for doing so(eg. index finger, middle finger pinch). The unawareness of the interactable area of the objects seemed to be a significant factor influencing how the gestures were performed. Overall, participants leaned towards the suitability of multi-touch gestures for parameter control in general, and confirmed its usefulness for multi-parametric interfaces. As one user commented:

"*Actually I prefer the one-finger gestures for being more intuitive and precise, but also see the benefit of multi-touch in reducing the need for additional buttons on the screen, leaving the interface very clean, calm and appealing.*"

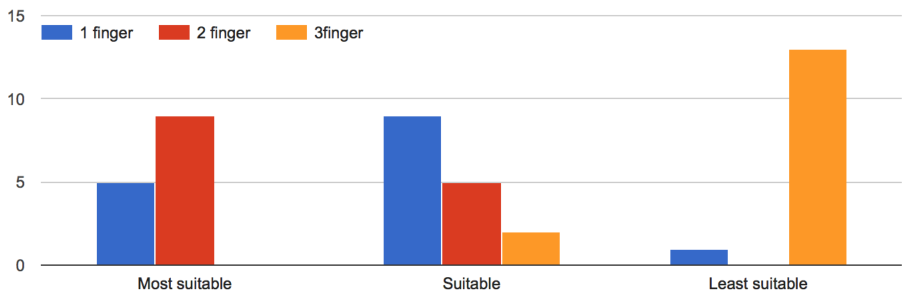

Figure 7.1: Distribution of answers for "In what order would you sort the three interactions, according to their suitability for parameter control?".

7.3.3 GUI and Visual Feedback

Participants found the design of the interface fitting to the application and aesthetically pleasing, but the lacked of feedback also highlighted by the evaluation. The most significant issue was related to the lack of feedback on the parameter limits and on the clear interactable area of the objects. Participants often tried to increase or decrease the parameters outside of their range, causing minor confusion. The performance of the interaction were influenced by users not being aware of what area belongs to the objects, as its not clearly visualised. This lead to errors in registering touch points, and therefore gestures. Other times, participants restricted their actions to the size of the rendered object, causing difficulties in performing actions.

Overlapping objects also disturbed the fluency of the interactions, and some participants highlighted the need for more visual information about the timing and sequencing of the oscillators.

Chapter 8

Discussion

The evaluation aimed to answer the posed research questions, to provide feedback for future development of *Posc*, and to potentially support gesture design for multi-parametric interfaces and mobile instruments. The evaluation was focusing on two main aspects, the users' gesture preference and the GUI design, summarised in the two research questions, discussed below.

Is vertex, polygon, or gesture based interaction more preferred, than the others for parameter tweaking?

The evaluation indicated, that participants have preference for the two finger *anchor* gesture, corresponding to vertex based processing frameworks. This question aimed to get a better understanding on preferred complexity for parameter control in musical applications, which is different in terms of input DOFs for each proposed gesture. Furthermore, focused on assessing the suitability of multi-touch gestures in general, for multi-parametric control, however, it is important to note, that it is a context specific finding, influenced by several potential biasing factors.

Not having a clear definition a target group, at such a small sample size(14), can significantly bias the findings. Most participants were familiar with musical interfaces, although, some were novices in many aspects. Similarly, the large screen size and the tabletop use of a touch screen, is a completely novel, to those who are accustomed to smaller hand held screen interactions. Unfamiliarity with fine multi-touch gestures also biasing the preference towards more simple and more common interactions, such as the *anchor*, which is very similar to the widely popular pinch to zoom/scale gesture. This consideration is also supported by the observation, as many participants tried to perform the proposed multi-touch gestures without their thumb. This is directly influencing the comfort, the accuracy and the quality of control. Participants using two hands in order to get a more comfortable grasp, was also common. The usability issues, regarding the gesture registration, imply further bias on gesture preference. Failure in gesture registration could render certain interactions unreliable, to some. This problem can be clearly accounted to the lack of visual feedback marking the interactable area. The list of tasks helped users to get an understanding about the system's functionality, but most likely, it was not able to provide proficiency in using complex gestures. It is believed the that the closer evaluation of three finger gestures in longitudinal method, could yield different results. The macOS operating system is utilising three and four finger gestures as well, arguing the ability to learn to use such. Ultimately, the findings suggest, that the increase in number of fingers renders the gesture less convenient and more cumbersome to produce, but more input DOFs could lead to more accurate control.

Can a solely symbolic GUI convey satisfactory amount information about polygonal waveform synthesis?

Yes, users were generally satisfied with the GUI design, and they reflected, that the visual feedback provided sufficient information in most cases. Similar GUIs have been designed for successful applications before, listed in Chapter 5, serving as the main inspiration. The goal of this evaluation was to investigate how suitable symbolic GUI is, for a completely novel application. The usability test highlighted that the parameter limits can be just as important as the parameter changes and states in terms of feedback. Participants lack of knowledge about the parameter limits conflicted with their mental model. Observations often pointed out actions which meant to increase or decrease a certain parameter beyond its limits. Although, it is believed, that with further refinement of the GUI design this can be avoided. Other feedback regarding the GUI was related to timing and sequencing capabilities. These yield lesser importance from the perspective of the underlying algorithm, but contribute to future design considerations. The effects of learning are also notable in this aspect of the evaluation. The participants had no problem completing the task list, and getting an overview on the functions, but some required time to understand the relationship between interactions, visuals, and sound. This also supports longer evaluations.

8.1 Summary and Future Works

Based on the feedback acquired from the evaluation, it is possible to outline the next iteration in the design cycle, and potential consideration for the next evaluation as well.

The gesture registration proved to be one of the most significant aspect of gesture design, along side with comfort and the convenience of the interactions. The findings support the two-finger gesture for parametric control. Possibility to control pitch and timing discretely, must be added to facilitate the musical application. To improve the interaction with GUI elements, overlap should be avoided, and areas of interaction must be clearly indicated. Visual feedback on parameters proved equally crucial for parameter change, parameter state, and parameter range. Lacking feedback on parameter range can significantly interfere with the construction of mental models. The waveform of the synthesised signal provides important information about the parameters, suggesting the implementation of an oscilloscope. Furthermore, the the authors of the original paper, propose modulation, cross-modulation and additional wave shaping, for future technical direction for the algorithm[15].

Many participants wanted to sustain the note while changing the parameters, and the built in sequencer received the most negative feedback. For keeping the simplicity, the redesign towards a performance oriented, virtual instrument is considered. This would potentially give a better application for multi-parametric control and musical expression. MIDI compatibility could further enhance the versatility, and integratibility to existing setups.

Both for designing and evaluating future a next iteration, a more rigid target group has to be defined. The next evaluation would also benefit from a more detailed demographics information gathering, to be able to triangulate differences, based on personal background. Longitudinal evaluation could potentially eliminate the learning curve, enabling the investigation of more complex gestures. Finally, complexity based on higher number of output DOFs and more sophisticated relationship analyses, could be the subject for future evaluations.

Chapter 9

Conclusion

This thesis presented the collection of research supporting the design of a mobile based, multi-touch DMI. Along with a novel synthesis technique, the findings were demonstrated in the design, development, distribution and evaluation of a mobile instrument. The evaluation was investigating gesture preference for parameter control, and the design of symbolic GUIs. The findings highlighted several important design requirement for further development, and promoting future research in the application of multi-touch gestures for multi-parametric control and mobile instruments.

Bibliography

[1] Nicholas Arner. "Investigation of the use of Multi- Touch Gestures in Music Interaction". MA thesis. 2013.

[2] Georg Essl, Ge Wang, and Michael Rohs. "Developments and Challenges turning Mobile Phones into Generic Music Performance Platforms". In: *Proceedings of the Mobile Music Workshop* (2008).

[3] Atau Tanaka et al. "A Survey and Thematic Analysis Approach as Input to the Design of Mobile Music GUIs". In: *NIME Proceedings* (2012).

[4] Thor Kell and Marcelo M Wanderley. "A Quantitative Review of Mappings in Musical iOS Applications". In: *SMC Proceedings* (2013).

[5] Thor Kell and Marcelo M Wanderley. "A High-Level Review of Mappings in Musical iOS Applications". In: *ICMC Proceedings* (2014).

[6] Mehdi Khosrow-Pour, ed. *Encyclopedia of Information Science and Technology. Mobile Music Interfaces Evaluation, Third Edition:* IGI Global, 2015, pp. 5686–5702.

[7] *AudioUnit - Apple Developer Documentation*. URL: `https://developer.apple.com/documentation/audiounit?language=objc` (visited on 08/31/2018).

[8] *Audiobus: Live, app-to-app audio.* URL: `https://audiob.us/` (visited on 08/31/2018).

[9] *Ableton Link: Connect music making apps with Ableton Live | Ableton*. URL: `https://www.ableton.com/en/link/` (visited on 08/31/2018).

[10] Patrick McGlynn et al. "Recontextualizing the Multi-touch Surface". In: *NIME Proceedings* (2012).

[11] Dalia El-Shimy and Jeremy R. Cooperstock. "User-Driven Techniques for the Design and Evaluation of New Musical Interfaces". In: *Computer Music Journal* 40.2 (2016), pp. 35–46.

[12] Perry Cook. "Principles for Designing Computer Music Controllers". In: *NIME Proceedings* (2001).

[13] Nicola Orio, Norbert Schnell, and Marcelo M Wanderley. "Input Devices for Musical Expression: Borrowing Tools from HCI". In: *NIME Proceedings* (2001).

[14] Marcelo Mortensen Wanderley and Nicola Orio. "Evaluation of Input Devices for Musical Expression: Borrowing Tools from HCI". In: *Computer Music Journal* 26.3 (2002), pp. 62–76.

[15] Christoph Hohnerlein, Maximilian Rest, and Julius O Smith III. "Continuous Order Polygonal Waveform Synthesis". In: *ICMC Proceedings* (2016).

[16] Christoph Hohnerlein, Maximilian Rest, and Julian D Parker. "Efficient Antialiasing of a Complex Polygonal Oscillator". In: *DAFx Proceedings* (2017).

[17] Chris Carlson and Ge Wang. "Borderlands: An Audiovisual Interface for Granular Synthesis". In: *NIME Proceedings* (2012).

[18] Sergi Jorda et al. "The reacTable: exploring the synergy between live music performance and tabletop tangible interfaces". In: *Proceedings of the TEI Conference on Tangible and Embeded Interaction* (2007).

[19] Sergi Jorda. "FMOL: Toward User-Friendly, Sophisticated New Musical Instruments". In: *Computer Music Journal* 26.3 (2002), pp. 23–39.

[20] *NodeBeat*. URL: `http://nodebeat.com/` (visited on 08/17/2018).

[21] Andrey Smirnov. *Sound in Z: Experiments in Sound and Electronic Music in Early 20th Century Russia*. Koenig, 2013, pp. 175–236.

[22] *A Brief History of Optical Sound Synthesis*. URL: `http://jiyounkang.com/wordpress/index.php/archives/1142` (visited on 08/14/2018).

[23] Thomas Y. Levin. "Tones from out of Nowhere: Rudolph Pfenninger and the Archaeology of Synthetic Sound". In: *Grey Room* 12 (2003), pp. 32–79.

[24] Jo Hutton. "Daphne Oram: innovator, writer and composer". In: *Organised Sound* 8 (2003), pp. 49–56.

[25] Peter Manning. "The Oramics Machine: From vision to reality". In: *Organised Sound* 17 (2012), pp. 137–147.

[26] *About Daphne*. URL: `http://daphneoram.org/daphne/` (visited on 08/15/2018).

[27] Yasuhiro Mitsuhashi. "Audio Signal Synthesis by Functions of Two Variables". In: *Journal of the Audio Engineering Society* 30.10 (1982), pp. 701–706.

[28] Bill Verplank, Max Mathews, and Rob Shaw. "Scanned synthesis". In: *The Journal of the Acoustical Society of America* 109.5 (2001), pp. 2400–2400.

[29] Dominik Chapman and Dr Mick Grierson. "N-gon Waves - Audio Applications of the Geometry of Regular Polygons in the Time Domain". In: *ICMC Proceedings* (2014).

[30] *NIME | International Conference on New Interfaces for Musical Expression*. URL: http://www.nime.org/ (visited on 08/18/2018).

[31] Marie Gonzalez-Inostroza, Alvaro Sylleros, and Rodrigo F Cadiz. "Understanding Interaction In Musical Multi-Touch Surfaces". In: *ICMC Proceedings* (2017).

[32] Fabio Morreale and Antonella De Angeli. "Musical Interface Design: An Experience-oriented Framework". In: *NIME Proceedings* (2014).

[33] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. "User-defined Gestures for Surface Computing". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2009).

[34] *Ocarina by Smule - Experience Social Music*. URL: https://www.smule.com/ocarina/original (visited on 08/18/2018).

[35] Ge Wang. "Principles of Visual Design for Computer Music". In: *ICMC Proceedings* (2014).

[36] James Mooney. "Frameworks and affordances: Understanding the tools of music-making". In: *Journal of Music, Technology and Education* 3.2 (2011), pp. 141–154.

[37] Thor Magnusson. "Designing Constraints: Composing and Performing with Digital Musical Systems". In: *Computer Music Journal* 34.4 (2010), pp. 62–73.

[38] Sergi Jorda. "Digital Instruments and Players Part I : Efficiency and Apprenticeship". In: *NIME Proceedings* (2004).

[39] Sergi Jorda. "Digital Instruments and Players Part II : Diversity, Freedom and Control". In: *ICMC Proceedings* (2004).

[40] M.S. Hancock et al. "Rotation and Translation Mechanisms for Tabletop Interaction". In: *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems(TABLETOP '06)* (2006).

[41] Katie Wilkie, Simon Holland, and Paul Mulholland. "What Can the Language of Musicians Tell Us about Music Interaction Design?" In: *Computer Music Journal* 34.4 (2010), pp. 34–48.

[42] Andy Hunt, Marcelo M Wanderley, and Ross Kirk. "Towards a Model for Instrumental Mapping in Expert Musical Interaction". In: *ICMC Proceedings* (2000).

[43] Andy Hunt and Ross Kirk. *Trends in Gestural Control of Music. Mapping Strategies for Musical Performance*. IRCAM, 2000, pp. 231–258.

[44] M.M. Wanderley and P. Depalle. "Gestural Control of Sound Synthesis". In: *IEEE Proceedings* 92.4 (2004), pp. 632–644.

[45] Andrew Johnston. "Beyond Evaluation: Linking Practice and Theory in New Musical Interface Design". In: *NIME Proceedings* (2011).

[46] Joel Chadabe. "Interactive Composing An Overview". In: 8.1 (1984), pp. 22–27.

[47] Sergi Jorda, Carles F. Julia, and Daniel Gallardo. "Interactive surfaces and tangibles". In: *XRDS: Crossroads, The ACM Magazine for Students* 16.4 (2010).

[48] Meredith Ringel Morris, Jacob O Wobbrock, and Andrew D Wilson. "Understanding Users' Preferences for Surface Gestures". In: *Graphics Interface Proceedings* (2010).

[49] Halla B. Olafsdottir, Theophanis Tsandilas, and Caroline Appert. "Prospective Motor Control on Tabletops: Planning Grasp for Multitouch Interaction". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2014).

[50] Mike Wu and Ravin Balakrishnan. "Multi-Finger and Whole Hand Gestural Interaction Techniques for Multi-User Tabletop Displays". In: *Proceedings of the 16th annual ACM symposium on User Interface Software and Technology* (2003).

[51] M. Wu et al. "Gesture Registration, Relaxation, and Reuse for Multi-Point Direct-Touch Surfaces". In: *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems(TABLETOP '06)* (2006).

[52] Russell Kruger et al. "Fluid Integration of Rotation and Translation". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2005).

[53] Kevin Schlei. "Relationship-Based Instrument Mapping of Multi-Point Data Streams Using a Trackpad Interface". In: *NIME Proceedings* (2010).

[54] Liwen Xu. *Multitouch and Gesture: A Literature Review of Multitouch and Gesture*. 2015. URL: `http://www.cs.toronto.edu/~xuliwenx/files/HCI_Research_Paper.pdf` (visited on 08/19/2018).

[55] Blake Johnston, Owen Vallis, and Ajay Kapur. "A Comparative User Study of Two Methods of Control on a Multi-Touch Surface for Musical Expression". In: *NIME Proceedings* (2012).

[56] Victor Lazzarini. *Computer Music Instruments*. Springer International, 2017, pp. 275–294.

[57] *Reactable Mobile*. URL: `http://reactable.com/mobile/` (visited on 08/17/2018).

[58] Joshua Peschke and Axel Berndt. "The Geometric Oscillator: Sound Synthesis with Cyclic Shapes". In: *Proceedings of Audio Mostly Conference* (2017).

[59] *DIN*. URL: `https://dinisnoise.org/` (visited on 08/17/2018).

[60] *TC-11 - Bit Shape*. URL: `http://www.bitshapesoftware.com/instruments/tc-11/` (visited on 08/17/2018).

[61] *fiddl*. URL: `https://itunes.apple.com/us/app/fiddl/id1320021639?mt=8` (visited on 08/24/2018).

[62] Michael Castanieto et al. "Fiddl: Evaluation of a Smartphone Based Application Designed for a Music Workshop for Children". In: *SMC Proceedings* (2018).

[63] *Borderlands*. URL: `http://www.borderlands-granular.com/app/` (visited on 08/24/2018).

[64] *JUCE*. URL: `https://juce.com/` (visited on 08/26/2018).

[65] *Posc*. URL: `https://itunes.apple.com/us/app/posc/id1419992583?mt=8` (visited on 08/26/2018).

[66] Fabian Esqueda, Stefan Bilbao, and Vesa Valimaki. "Aliasing Reduction in Clipped Signals". In: *IEEE Transactions on Signal Processing* 64.20 (2016), pp. 5255–5267.

[67] Fabian Esqueda, Vesa Valimaki, and Stefan Bilbao. "Rounding Corners with BLAMP". In: *DAFx Proceedings* (2016).

[68] Steven Gelineck and Stefania Serafin. "Longitudinal Evaluation of the Integration of Digital Musical Instruments into Existing Compositional Work Processes". In: *Journal of New Music Research* 41.3 (2012), pp. 259–276.

[69] Jeronimo Barbosa et al. "What Does "Evaluation" Mean for the NIME Community". In: *NIME Proceedings* (2015).

[70] Dom Brown, Chris Nash, and Tom Mitchell. "A User Experience Review of Music Interaction Evaluations". In: *NIME Proceedings* (2017).

Appendix A

Post-test Questionnaire

A.1 Section 1. Tutorial

Before you proceed... please complete the following tasks listed below, while commenting you actions:

1. double tap on the screen to create an oscillator
2. reposition the oscillator by tapping and moving it around
3. tap and hold to enter into "idle mode", then move your finger in a circular motion
4. tap with 2 fingers and pinch
5. tap with 2 fingers and rotate
6. tap with 3 fingers and pinch
7. tap with 3 fingers and rotate
8. drag the oscillator to the bottom of the screen to delete it

After that... feel free to experiment with the application, while you keep commenting your actions.

A.2 Section 2. Background

- Age
- Experience with electronic musical interfaces or controllers(Familiarity with controlling audio with knobs, faders sliders, etc.)
- Experience with mobile music apps

A.3 Section 3. Usability

To what level do you agree with the following statements? (1 - disagree, 5 -agree)

- I found the application easy to learn
- I found the interaction intuitve
- I felt confused during the test
- I found the control of the parameters accurate/precise
- My actions were translated as I have expected
- I felt in control of the application
- I found the interface responsive
- I found the graphical user interface informative
- The application have lacked feedback
- I found the interface aesthetically pleasing
- I found the free experimentation engaging
- I felt that there is more to explore
- I would want to play it it more

A.4 Section 4. Gesture preference

How would you rate your experience with the parameter tweaking with touch gestures?(Comparing one(circular movement), two and three finger interactions.)

- Which of the three interactions did you find the most intuitve?
- Which of the three interactions did you find the easiest to perform?
- Which of the three interactions did you find the most comfortable to perform?
- Which of the three interactions did you find the most accurate?
- In what order would you sort the three interactions, according to their suitability for parameter control?
- Please elaborate on your previous answers.(e.g. argument for preference)

A.5 Section 5. Final remarks

- Did you find the graphical user interface fitting for the application?
- Could you see yourself controlling sound parameters with multi-touch gestures?
- Any other comments:

Appendix B

Participant time sheet

Participant	Time spent on task list (mm:ss)	Time spent experimenting (mm:ss)
#1	06:30	05:30
#2	01:52	00:37
#3	01:47	08:10
#4	00:57	01:32
#5	02:39	05:12
#6	03:12	02:54
#7	04:13	11:05
#8	03:44	04:05
#9	02:25	04:36
#10	05:30	05:14
#11	02:41	05:12
#12	02:58	06:56
#13	04:30	05:11
#14	02:10	03:31
Mean	03:13	04:59
STD	01:32	02:38
Min	00:57	00:37
Max	06:30	11:05