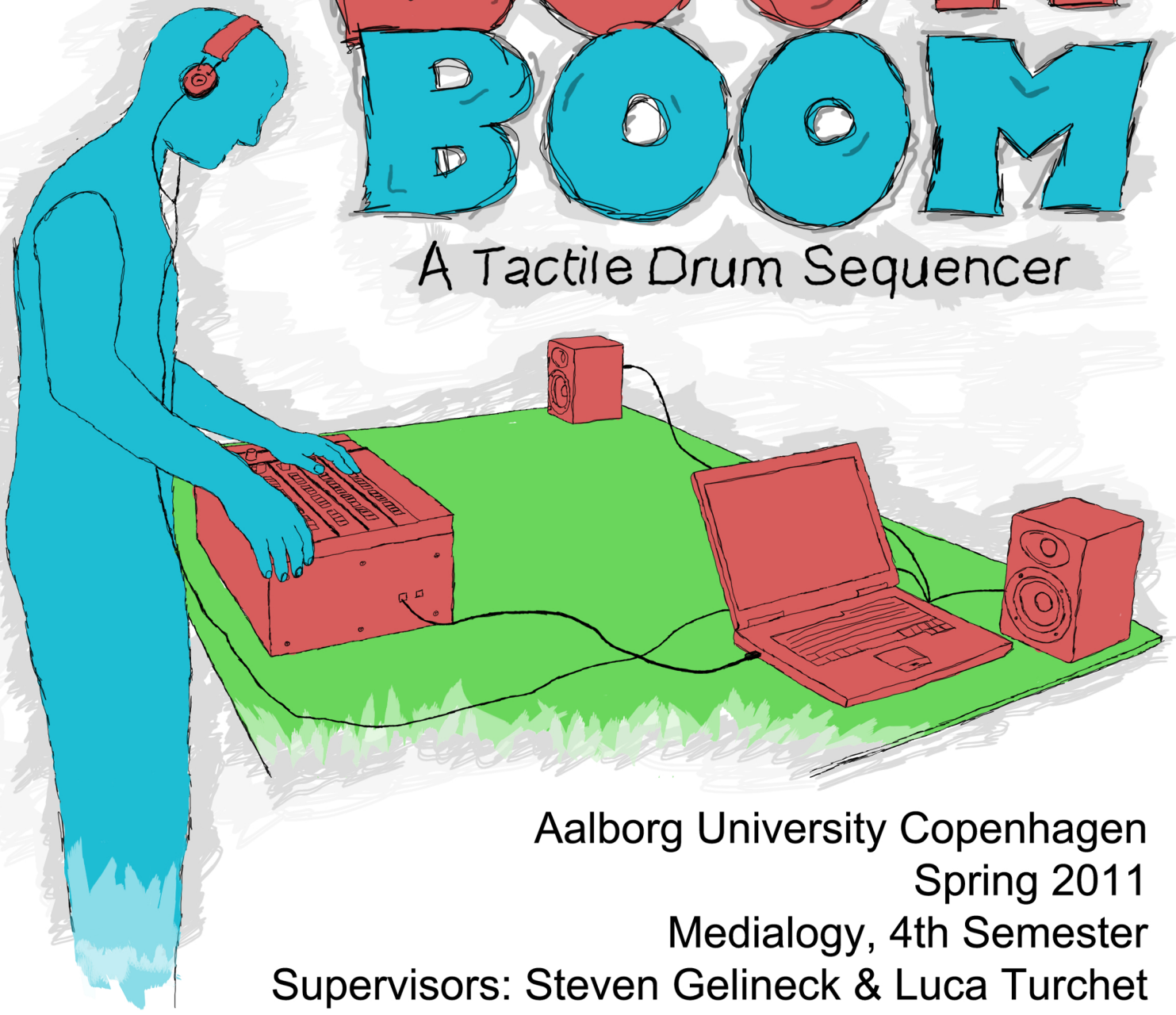
Group 403

Alexander Birke
Rune Svante Stenstrøm
Jan Olesen
Marcel Schmidt

TactiBOOM BOOM

A Tactile Drum Sequencer



Aalborg University Copenhagen
Spring 2011
Medialogy, 4th Semester
Supervisors: Steven Gelineck & Luca Turchet

Semester: **4th**
Title: Tacti BOOM BOOM – A Tactile Drum Sequencer

Project period:
February 2011 – May 2011

Semester theme:
Interaction Design – Sound and Sensors

Supervisor:
Steven Gelineck

Co-Supervisor:
Luca Turchet

Project group no.:
k11ml403

Members:

Alexander Birke

Jan Olesen

Marcel Schmidt

Rune Svante Stenstrøm

Aalborg University Copenhagen
Lautrupvang 15, 2750 Ballerup,
Denmark

Secretary: Lisbeth Nykjaer
Phone: (+45)99402470
lny@create.aau.dk

Synopsis:

Even though music plays a large role in many blind people's lives they have many hardships when it comes to creating and playing music. This project therefore seeks to create an interface that solves a problem in this area.

One area where further improvements can be made is in the programming of drum sequences. It is therefore chosen to see if a tangible interface can benefit blind users in this task since this type of interface is assumed to have several advantages over existing dialogue driven alternatives.

Relevant HCI theories regarding tangible interfaces and interfaces for the blind are researched along with haptic gestalt principles. Rhythm theory and an overview of what constitutes a drum machine is also researched.

In the design phase it is sought trough usability testing to find the best way to convey the different parts of a drum machine's interface.

The project concludes with a test of the prototype with blind musicians. From this test it can be concluded that the prototype meets the stated goals of usefulness and satisfaction but that some aspects of the product can be improved.

Preface

Reader's Prerequisites

It is assumed that the reader has the same knowledge as a student at medialogy that has completed the fourth semester. Furthermore are different text formats used to make it easy to distinguish between the different methods used in the writing of the report. An explanation of these is given in the table below:

Method	Format	Description	Example
Quotations	""	All direct quotations from source material is framed by quotation marks and followed by source reference.	*"An example of a quotation"*
References	[number]	All references are marked by a number.	[42]
Footnotes	*superscript*	Explanation of difficult words or relevant additional information.	[4]

Word list

Following is a list of words encountered in the report that the reader should know the meaning of before further reading.

Blind
: With blind is meant the medical definition "totally blind". An explanation of this definition can be found in the preanalysis, section *"Defining The Target Group"*.

Haptic Perception
: Haptic perception theory seeks to describe how humans use the sense of touch to gain information about the world.

Tactile Feedback
: Feedback provided when you touch something. It can both be the texture of the material being touched but also its resistance to pressure can be described as tactile feedback.

Haptic Feedback and Haptics
: Most commonly is feedback from an interface trough kinaesthetic feedback denoted haptic feedback but the term vibrotactile feedback is also used. Technologies that provides haptic feedback are called haptics.

Reader's Guide

The report starts with an **introduction** that outlines the overall theme of the report, and the motivation behind the project. To explore the problem space further the **initial problem statement** below is formulated:

> *How can an interface be created that helps blind people create and produce music?*

The initial problem statement is then sought to be answered in the **pre-analysis** by first seeking a definition of the target group, what types of general computer interfaces that exist today to assist the blind. furthermore it is also researched what posbilities the blind have when it comes to music creation. Both commercial solutions and relevant research is described in this section Lastly an interview is conducted to further explore these subjects. The pre-analysis culminates in the **final problem statement**:

> *How can a satisfying tangible interface with a sufficient degree of utility be created that enables blind people to create drum sequences?*

The **analysis** covers relevant HCI theory regarding tangible interfaces and interfaces for the blind. Haptic perception and haptic gestalt principles are also researched. An overview of what constitutes a drum sequencer and general rhythm theory is also provided along with an explanation of how sound can be panned. After a section describing how to test with impaired people the analysis concludes with a list of **product requirements**.

In the **design** chapter, the process of meeting these requirements are described. Several formative tests used in the design of the product are explained here. The chapter ends with an overview of the final design.

This process of implementing this final design is then described in the **implementation** chapter where the different parts of the prototype are covered. Appendix F, which covers the relevant information regarding to sensors technology, accompanies the implementation.

The implementation leads to the final **test** chapter were the summative test used to assess the prototype's success is described along with the result of the test. This is followed up by a **discussion** of the test results.

The report culminates in an overall **conclusion** as well as a discussion of **further perspectives** for the project and the prototype.

The Product

On the DVD accompanying this report can two videos demonstrating the use of the product be found. One titled "Solo Playing" demonstrates the use of the product by a blind folded proxy user. The other titled "Collaborative Playing" shows how two non-blindfolded proxy users uses the product in a collaborative way.

Recognition and Appreciation

We would like to thank the Danish Institute of Blind and Visually Impaired and the test participants that helped us conduct the final summative test of the prototype.

Contents

Chapter 1

Introduction

Missing one of the five senses is for most people unthinkable. Imagine to live your life without being able to see. Even simple tasks, like going to the supermarket, taking the bus, or taking out the trash becomes an enormous challenge.
According to the AMA Guides, the American Medical Association's guide to assess disability, a person with total loss of vision, stated as a 100% visual impairment, has a whole person impairment of 85% [9]. Even with this high level of impairment most blind people can survive in the modern world without immense problems. One reason for that is that blind people compensate their lack of vision with enhancement of their other senses. Research has shown that totally blind people, depending on the age they became blind, can hear better than people without visual impairment [13]. Another reason is that there exist many supportive facilities, which help disabled people to navigate through every day life and eases everyday exercises. Examples could be screen readers that helps blind people to use computers, assistance dogs that help blind people to navigate in physical environment, and the Braille system that enables blind persons to read.
Unfortunately there are some areas where further development could be reached. One area is the creation and production of music. Large commercial composing programs does not provide the kind of interaction that blind persons require in order to use the software. Additional plugins seems to be needed before such programs can be utilized. When it comes to learn how to learn and play instruments different audio based learning materials has been developed [1]. However more research into blind peoples use of music instruments and how they produce music could reveal areas where further improvements can be made.
Since many blind people show an interest in music, we believe that they should have the same possibilities in this regard as normal sighted do. In the context of medialogy, the following report aims to target a specific issue in this problem space.
To localize an unsolved issue in regards to blind people's creation of music it is necessary to gain a deeper understanding of the problem space. Both blind people's use of traditional instruments and traditional ways of composing will be researched along with the new ways to create music that has been emerging since the dawn of the personal computer. The goal of the report will be to develop an interface prototype that will address a specific problem in the problem space an assess its level of success. The initial problem statement is therefore:

How can an interface be created that helps blind people create and produce music?

In the next chapter, the preanalysis, this question will be answered.

[1] `http://www.musicfortheblind.com/`

Chapter 2

Pre-analysis

In order to answer the initial problem statement further analysis of the problem space will be needed. Trough research it is established that three key areas are necessary to be answered.
First it will be necessary to get an understanding of the target group and how their disability affect them. This will also include the need to research a proper definition of what blindness means since this will be the primary boundary for the target group.
Secondly, since the scope of this project is to develop some type of electronic interface it will be relevant to gain an understanding of the types of interactions with a computer that is supported for visually impaired people.
The third and final area will be to understand how blind people play and compose music but most importantly, what problems and hardships they encounter in doing so as this will form the foundation of the final problem statement. This will also include a study into what types of commercial products that already exist to help blind people create music. It will also be investigated what solutions are used in Denmark.
Since the target group of this project will be persons with a visual impairment, interfaces that uses the modality of touch are obvious to investigate. Therefore will tangible interfaces and instruments also be investigated.
In the next section the target group will be further researched.

2.1 Defining the Target Group

This section will clarify the term visual impaired people to create an understanding of the disability and how it is defined.
The Danish medical definition for being visual impaired is to have a visual impairment of 6/60 or less on the best eye. 6/60 is understood as what a normal sighted person can see on 60 meters, a person with 6/60 can see within a distance of 6 meters. Between 6/60 and 0/60 people are categorized in three sections, socially blind, practically blind and totally blind as shown on table 2.1 [4].

Description	Visual Impairment
Low Vision	6/18 to 6/60
Socially Blind	6/60 to 1/60
Practically Blind	6/18 to sense of light with projection
Totally Blind	No sense of light & sense of light without projection

Table 2.1: Danish definition of visual impairment(Our translation)

To get ones degree of visual impairment an ophthalmologist conducts tests to assess visual acuity, visual field, vision in the dark and how they work together.
Another definition used when talking about visual impaired people is a social aspect. It is said that visual disabilities depends on relations and situation. Hence, a visual disability is not a handicap but depends on how the person with the disability behaves in a social situation. A blind person can be well integrated in society if self-engagement is high. On the other hand, if a blind person feels like giving up, it becomes a handicap, and will therefore require help one way or another. Therefore one of the big visions in the Danish society is to figure out how to integrate visually impaired people in the best possible way. Whether or not people with a visual impairment are well integrated can be related to when they became visually impaired. The later they get the impairment the higher the risk for social isolation becomes. The graph on figure 2.1 shows in what age Danish people get their visual impairment [4].

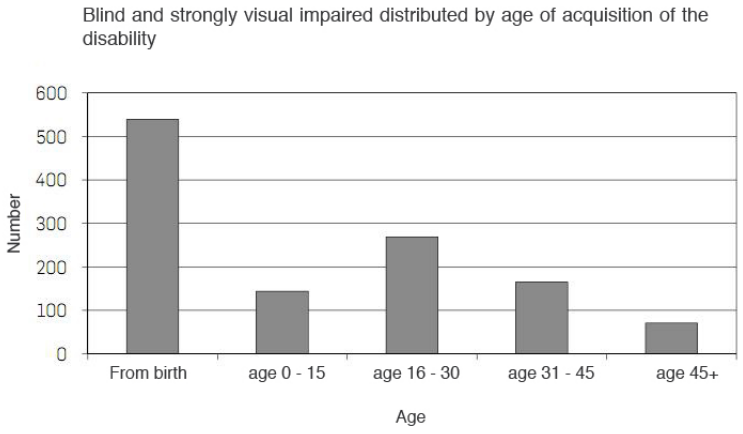


Figure 2.1: Bar chart showing the distribution of when blind and visually impaired got their impairment

People with visual impairment though seek social contact with other people. Mostly they are together with people of the same disability, because they have more in common regarding of what they are able to do. When it comes to how they use their spare time, they get more together in social relations than normal sighted people. They also tend to be more engaged in associations as illustrated by table 2.2 [4].

	People with no visual impairment	people with a strong visual impairment	blind	$_x$2-value; P-value
Educational activities in spare time	13	21	30	69,49; <0,0001
Engaged in associations	57	54	58	2,81; <0,24
Voluntary work	25	34	43	55,31; <0,0001
Base of calculations	1.780 - 1.809	973 - 1.022	323 - 340	

Table 2.2: Percentage of people with no visual impairment, visual impairment, and blind people that participate in spare time education, associations, and volunteer work. (Our translation).

There are no actual numbers of people with visual impairment in Denmark, but according to [4] there are 52.000 out of 3,5 million in the age group 16-64 with visual impairments. 27.000 out of the 52.000 are socially blind, 14.000 practically blind and 12.000 are totally blind.

This section has clarified the term blind and it has become clear that it covers three different categories of visual disabilities. This information is important when choosing a specific problem and target group for the final problem statement since socially blind people and totally blind people could have very different problems and needs. The next section will look into computer interfaces for visual impaired people.

2.2 Computer interfaces for visually impaired people

Since there are different categories of visual impairment there are also different solutions depending on the degree of the impairment.
When a visual impaired person has to choose a computer interface it depends on the degree of impairment and what the person feel comfortable with. If the person for instance is socially blind and therefore has some vision a screen magnifier, which is a piece of software that magnifies screen output, can be suitable solution. Most operating system have one implemented and there are also developed more advanced versions like Zoomtext by Aisquared which are designed specifically for visual impaired people [1]. Aisquared also develops other equipment for visual impaired people like keyboards with large prints.

Totally blind people have other needs, since they do not use the same reading and writing system as sighted people. Instead they use a system called the Braille system [2]. This system is based on touch reading and consists of raised dots that represent the normal alphabet (see figure 2.2) and can written by hand trough the use of a stylus and slate or a special type of paper.

[1] `http://www.aisquared.com/`
[2] The Braille system was invented by Louis Braille in 1825.

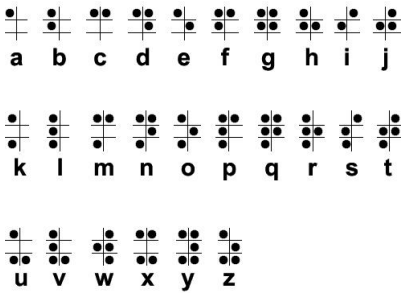


Figure 2.2: braille alphabet, there are also Brailles for numbers and special characters

This does not necessarily mean that they need a special keyboard when interacting with a computer but they do need a piece of software called a screen reader. One popular screen reader is JAWS[3]. A screen reader gives two kind of output to the user, one is a voice output of what is written on the screen, for instance if the user highlights a word or strikes a key. The other is provided through a Braille display. A Braille display is a piece of hardware that translate the written text on screen into Brailles(see figure 2.3). Some versions also offers a Braille keyboard for user input.

Figure 2.3: A Braille display raises the Braille letters through retractable dots

HCI models have also been developed to create guidelines for how to design interfaces for blind people. One such is the FBLIND framework. FBLIND stands for Framework for BLind user INterface Development and can be seen on figure 2.4. The framework seeks to incorporate some of the most widely accepted HCI models by designers and researchers, [29] and describe how they should be used in the context of blind users.

One part of the framework is the SBIO(Speech and Braille Input and Output) model. It describes how a blind user should be able to perform the tasks described in the Task model so they do not involve hand-eye coordination or simultaneously controlling more than one visual item at a time since they communicate with a computer in a serial manner and only interacts with one widget at a time 2.4. This forces the interaction to happen as a dialog between the blind person and the interface. The FBLIND framework thus seek to illustrate how all the popular models it incorporates are affected by the requirements of blind users.

[3] `http://www.freedomscientific.com/products/fs/jaws-product-page.aspl`

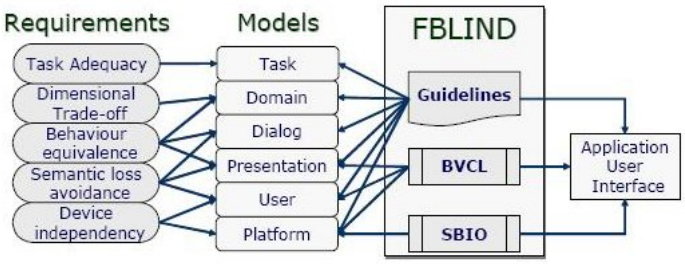


Figure 2.4: User Requirements, popular HCI models, and the FBLIND framework

2.3 How visually impaired people create music

This section will look into how visual impaired people learn to play, compose and produce music along with which solutions for these tasks that exist today.
First it will be covered how blind people play and learn traditional instruments. This will also cover composing trough the use of traditional musical notation.
Secondly, the new approaches for composing and producing music based on digital and electronic interfaces will be covered.

Traditional musical notation and instrumentation

There are many educational offers for blind people to learn to play, compose and produce music but how do these processes for visual impaired people differ from normal sighted people? It is obvious that totally blind people do not have the opportunity to read a book about learning to compose music or play an instrument. Instead they use specially made audio books. It is hard to find information about if some instruments are preferred and if some simply relies to much on visual feedback for some visual impaired people to use them.

Braille music notation

There has been formed a Braille system for music notation which is called Braille music that has been around for over 160 years (see figure 2.5). This gives visual impaired the opportunity to read and write music and *"form his own interpretation of the music, rather than simply parroting someone else's rendition from a recording"* [4]. There are also hardware and software that translate music notation into Braille music so sighted people and visual impaired people can teach each other to compose music or work together. A popular solution for this is is the GOODFEEL Braille Music Translator made by the company Dancing Dots[5]. It was the first software which allowed blind people to use the PC as support for their creativity. The GOODFEEL Braille Music Translator converts several kinds of music files to Braille.

[4] `http://www.blindmusicstudent.org/Articles/learning_teaching.htm`
[5] `http://www.dancingdots.com/main/goodfeel.htm`

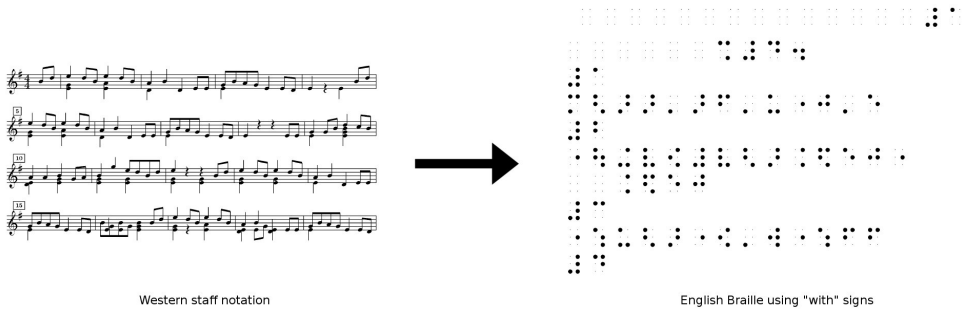


Figure 2.5: Conversion from western staff notation to English Braille

Digital solutions for music production

For persons with no visual impairment there are primary two solutions along with a combination of these two when it comes to record, compose, and produce music in an electronic manner. One is the traditional way with physical gear such as mixers, all sort of effect devices, and cables to connect the different devices in a studio environment as depicted on figure 2.6).

Figure 2.6: An example of a typical professional sound studio

The other way that has become more and more popular is the computer based approach where a software package called a DAW (Digital Audio Workstation) is used(see figure 2.7). This performs many of the typical functions of a big physical setup [28].

Figure 2.7: Examples of the graphical user interfaces of Digital Audio Workstations(DAW).

Besides a DAW it is common to use virtual instruments and effects which can be loaded in as plug-ins in the DAW. The DAW and its plugins can also be controlled trough devices called MIDI controllers to get a more familiar and natural way of interacting with the software, rather than using the mouse and computer keyboard. As it can be seen on figure 2.7 and 2.6 controlling both the physical and virtual systems depends on the modality of vision since most of the information from the interface is mediated trough visuals. This makes them inaccessible to visual impaired people if their condition is too severe. One advantage that the traditional studio setup has over DAW's is that it contains tangible elements like knobs, sliders, and so on. This is likely to make it more usable for visually impaired persons.

If the visual impaired person desires to use a DAW, and the visual impairment is so severe that it is not possible to use traditional graphical user interfaces, one of the only alternatives is apparently to use a DAW called Sonar by Cakewalk[6] . Dancing Dots, the company mentioned earlier in the report, have created a program called CakeTalking[7] that customizes the screen reader JAWS so visual impaired people interact with Cakewalk Sonar using key commands and auditive feedback. A mp3 presentation demo of the software can be found on the DVD.

In the audio presentation of how Caketalking works it appears that the software does not support the task of programming sequences, only how to import simple loops. This is a severe limitation since it is one of the main tasks of a DAW. A possible area of interest for this project could therefore be to develop a product that supports this task.

It's hard to find information about the software other than comments from musicians on Dancing Dots' webpage which claims it makes Cakewalk Sonar fully accessible for totally blind users. There is also a video with music artist Raul Midon which uses CakeTalking[8] and MIDI controllers to produce music where you get an idea of how it is used. However it does not go into depth or describe any limitations of the software but since it is posted on Cakewalk's web page it is likely be subject to bias.

There has also been conducted research that seeks to make visual content in audio applications

[6] http://www.cakewalk.com/products/sonar/
[7] http://www.dancingdots.com/prodesc/CakeTalkingForSONAR.htm
[8] http://blog.cakewalk.com/songwriter-raul-midon-produces-music-with-cake-talking-technology/

accessible trough the modality of touch.
One of these is The Moose, a haptic mouse that interfaces with Microsoft Windows's GUI [12]. A diagram of The Moose is depicted on figure 2.8.

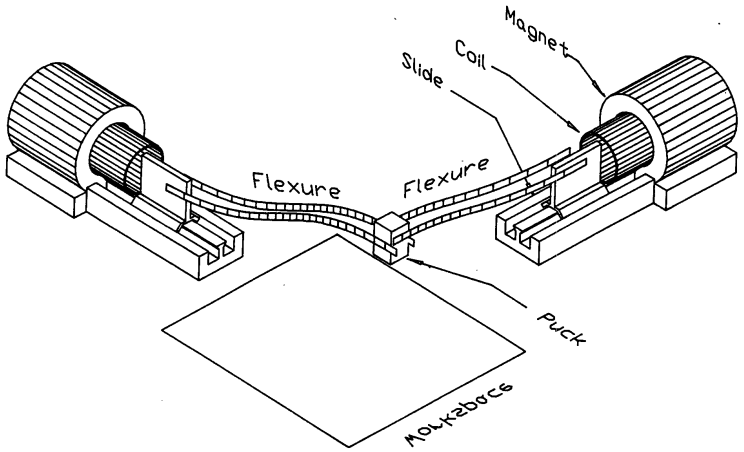


Figure 2.8: A diagram of The Moose. Two linear coil motors provide sensing of force applied by the user as well as haptic feedback

The main goal of The Moose was to provide a haptic way for blind sound engineers to interact with sound since the auditory channel of information already is taken up by the medium being edited. Moreover since the software developed for The Moose supported recognition of GUI elements it demonstrates that the GUI of a DAW likely can be made accessible trough haptics. It should be noted that The Moose is relatively old but never research is also available. For instance sought a similar project to solve the same task trough the use of a haptic knob and a vibrotactile mouse[8]. The above research demonstrates that information about audio that is normally conveyed trough visuals can be conveyed trough haptics instead. it is likely that haptics could be developed to work together with a DAWs interface.

MIDI controllers and Digital Instruments

When it comes to MIDI controllers it does not seem like there are made any commercial MIDI controllers specifically designed for visual impaired people. Often traditional MIDI controllers give the user feedback using LED lights or displays and use menus to set them up with different DAW packages. This makes them hard for visually impaired to use. On figure 2.9 can a broad selection of popular MIDI controllers be seen. These controllers have been selected to show that the described problem is something all types of MIDI controllers suffer from.

a) Roland TD-8 drum machine

b) Novation SL49MKII

c) ProjectMix

d) Novation Launchpad

Figure 2.9: A selection of popular MIDI controllers

The Roland TD-8 drum machine (figure 2.9.a) uses both a display and LED lights to convey information to the musician. Moreover the direction of the knobs on the device is only visualized by a strip of paint at the top of each knob. This makes it impossible to feel in what position the

knob is in. The ProjectMix mixer table from the company M-audio is interesting since it offers a bit of accommodation for blind musicians. The faders on the device are motorized and can therefore give the musician haptic feedback. The rest of the interface however, still relies on visual feedback from LED's and the LCD display located at the top of the device.
A new trend in MIDI controllers is to make them programmable by the musician. A popular example of this is the Launchpad from Novation that offers a grid of 8 x 8 programmable buttons. As with the other MIDI controllers described in this section, all feedback is given visually by what color the buttons are in, making it inaccessible to blind musicians. The MIDI keyboard depicted on figure 2.9.b is interesting since it is sold through Dancing Dots own homepage. One should therefore think it would be better suited for blind musicians than other MIDI keyboards, but as it can be seen on the picture, light is still used to convey the state of the instrument. This could indicate that there simply does not exist a commercial MIDI keyboard that accommodate blind people's needs. Since your own research into the matter was unsuccessful this seems to be the case. However there have been made some reaches to create a keyboard midi controller for blind people in a non-commercial context. This controller is described in the next section.

The tangible MIDI sequencer

Facing the same problem as this report, students from University of Mannheim have developed a tangible MIDI sequencer for visually impaired people [14]. To solve the problem that blind people have problems navigating a DAW's visual and complex interface, the purpose of the tangible sequencer is to steer the the system just with the use of the instrument. As can be seen on the figure 2.10 the chosen instrument is a MIDI keyboard.

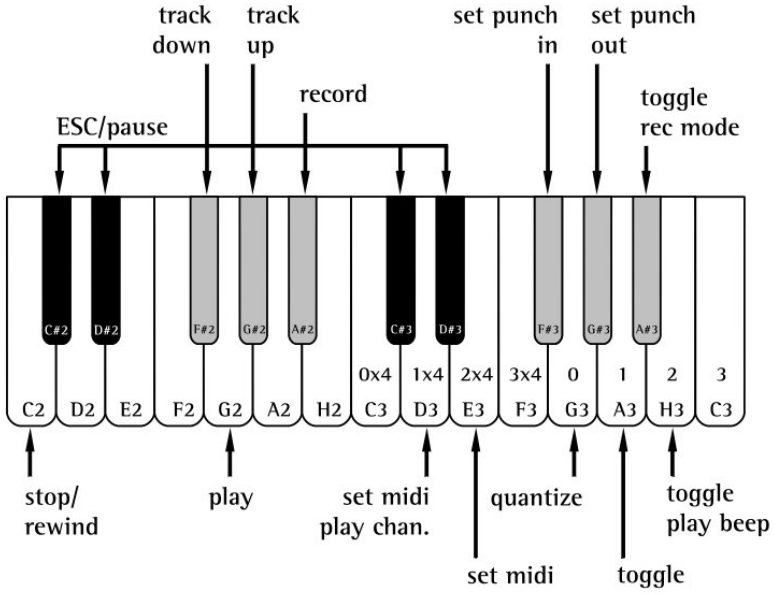


Figure 2.10: The keyboard can be used to control the sequencer (in command mode) and to play notes (in play mode)

By switching between command and play mode, the musician can edit and play music without taking his hands away from the keyboard. Auditory feedback is also provided to convey track numbers. By listening to a repetition of two different notes the musician can calculate the number of the track. A low note is interpreted as a multiple of four and a high one as a multiple of one.

10 would be decomposed as 2 x 4 + 2 x 1, which would occur as two low notes followed by two high notes. Nevertheless in some rare cases the user has to switch to the PC screen, for instance to enter the filename. For this purpose a basic graphical interface was designed, that can be easily navigated with a screen reader (see figure 2.11).

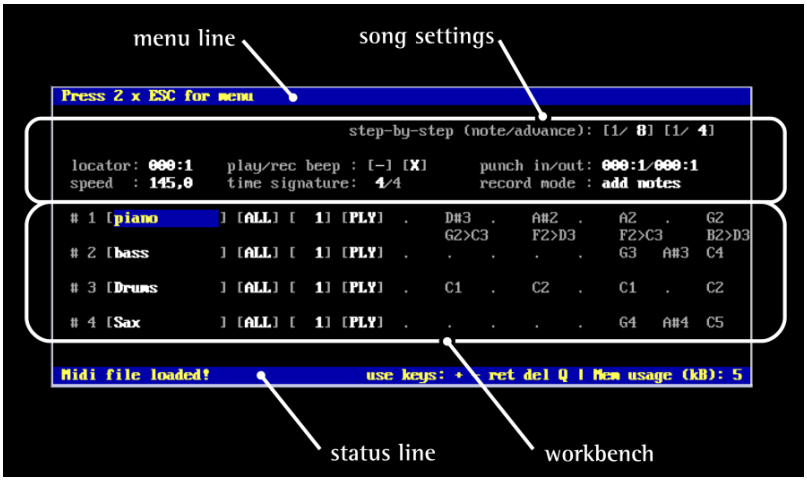


Figure 2.11: The user interface for the tangible MIDI sequencer is ASCII-only to make it easy to navigate for screen readers

Even though the tangible MIDI sequencer makes sequencer software more accessible for blind people, it still uses a physical interface which was designed for sighted people and can therefore lead to complications. For example, it is possible to operate the sequencer in two different modes, but the current mode can not be identified due to tangible feedback and therefore the occurrence of mode errors is quite likely [21].

Another area of research is how to improve the interaction between musician and digital music instrument by introducing the modality of touch to them. The main reason for this is that much of the state of a traditional instrument is conveyed to the musician by vibrations made by the sound created by the instrument and the physicality of the instrument itself. With physicality is simply meant that the instrument can be touched and manipulated in a physical manner which provides more natural feedback to the musician.
Since the modality of touch also seems to be a good medium for interaction between an instrument and a visually impaired person, it is chosen to look at research into tangible user interfaces and haptic feedback that has a connection to music.

The BeatBearing

The BeatBearing is an explorative design test bed that offers a tangible way of programming rhythm patterns [5]. It was created to test how the theory of embodied interaction and enactive design can inform the design of digital musical instruments. With embodied interaction is understood that interaction often is embodied, that the user has to take a physical action to partake in the interaction and that this physicality has implications both in what actions can be made and on the cognition of the user [10].Enactive design entails that perception is guided by the actions that are perceivable by the individual and that cognitive thought patterns(e.g. understanding) emerge from repeated sensorimotor patterns[10].
The aspects of embodiment and enactive design has only recently been researched in regards to

both general interfaces and digital musical instruments. Another goal of the BeatBearing is also to cover the gab between the theoretical foundation of design(and embodied design in particular) and explorative designs that currently exists.
The way patterns are created on the BeatBearing is by placing small metal balls on a 8x4 grid of holes. When a metal ball is placed in a hole it creates a connection between two terminals and thus forms a closed circuit. Under the grid is a CRT monitor placed that provides visual feedback. This can be seen on figure 2.12 where the red line shows what part of the pattern is currently being played. Coloured circles below the balls informs the user that the slot has indeed been triggered by the ball in it.

Figure 2.12: The BeatBearing

Several areas makes the BeatBearing interesting for this project. The fact that it is a tangible user interface means that it could very well support interaction for individuals belonging to all three categories of visual impairment described previously in the report. One concern with this interface however, could be that as a visual impaired user tries to feel the pattern of balls they are likely to be pushed out of their sockets, thus destroying the original pattern.
The other area that makes the design interesting is that it differs from other notable tangible user interfaces like the Reactable[18] in that it uses discrete points of interaction and utilizes physical constraints. Since visually impaired users most likely will not be able to see the device, the physical constraints and the discrete points of interaction will make it easier for them to understand and use the interface [21].

The DAMPER

One problem that exists with most electronic musical instruments is that they do not require the performer to produce an effort in order to use the instrument. With effort is meant that the interaction does not involve some sort of physical action to the same degree that a traditional instrument does. . With a traditional instrument it is quite clear that in order to produce a sound with a higher amplitude, the more force has to be applied in the action of using the instrument.
The DAMPER has been developed as a simple one-degree-of-freedom interface used to test how haptic feedback can lead to an effortful design of electronic instruments[6]. Another area that the DAMPER tried to explore was physicality, which simply means that the instrument exists in the physical world and can therefore be touched and felt. What is distinct about the DAMPER is that it is able to produce a dynamic resistance to the motion of the two handles it consists of. This is done with a Magneto-Rheological Fluid. This is a type of brake where the amount of resistance created by the brake can be adjusted by applying a current to a coil inside the brake. The angle between the two handles is obtained trough the use of a potentiometer used as a rotary sensor.

Figure 2.13: The DAMPER in the "edge-trimmer" configuration

The DAMPER has currently been tested in two configurations where the first, the edge-trimmer configuration is shown on figure 2.13. In this mode a force-sensing resistor is used as a button at the top of one of the handles. The angles between the two handles then determines the frequency of a square wave signal which amplitude is determined by the amount of pressure applied to the force-sensing sensor. This signal is then sent to the Magneto-Rheological Fluid brake. The result is that the more pressure applied by the user, the more force is required to move the handles. A novel aspect is that due to the signal being sent to the brake is in audible frequencies, sufficient vibrations are created so the instrument can be played acoustically if placed on a soundboard such as a desk.
In the other configuration is the resistance created by the brake proportional to the angle between the handles. The angle between the handles is used as input to a physical model of a string made as a MaxMSP patch.

Based on the research described above it seems like there are many different areas that could hold interesting problems. A summary of these problems is given in the overall conclusion on the preanalysis below.

2.4 Interview

To clarify if there is other problems with creating, composing or producing music other than what already has been researched in a Danish context, interviews with relevant people was sought to be conducted . It proved to be a tough task to recruit participants for this. Both members of a mailing list for visual impaired as well as a composer that had experience working with blind people was asked without luck. In the end one interview was secured.
The interviewed person (red. he) is an IT consultant from the Danish Institute For the Blind and Visually Impaired People, also called IBOS[9], and is the head responsible for helping blind high

[9]In Danish: "Det danske institut for blinde og svagtsynede"

school students with I under eduT related problems. He has no experience when it comes to producing music, but only knows what solutions that are available for blind musicians. He is blind himself.

The interview was conducted as an explorative interview [19], where the idea is to ask broad questions to get an idea of a problem. In this context open-ended questions are typically used. For example a question such as How was it to learn to play that instrument? was on the interview guide. By asking these kind of questions the interviewed person can answer freely, without the interviewer guiding them in some direction. The interview was organized as a semistructured interview[19], where questions was prepared before, but still open for discussion if new questions came in mind during the interview. This fits the initial exploration method because there can be problems not thought of by the interviewer. It is analysed by transcribing and find the relevant statements/problems said under the interview.
The interview can be found on the DVD, and the interview guide in appendix A.

The Interview

At the moment IBOS are suggesting blind people to use a program called Sibelius which is used to compose music with nodes. Sibelius works together with JAWS. When it comes to producing music on a computer he says that it is a very visual process that needs mouse-eye-coordination, and therefore is blind people's use of them limited to only a few programs where JAWS work. The problem they are facing at the moment is that the JAWS version for Sibelius is translated from the american version, and therefore the keyboard has to be set to american layout. This is annoying for the blind people when they are accustomed to use a danish keyboard.
He mentions a Danish node program used in the danish gymnasiums called PriMus[10], which blind people can't use. He suggests that we could make a plugin for JAWS so PriMus can be used by blind people.

This interview indicates that only node programs are used by Danish blind people and that there still room for improvements in this area. DAWs are not used since they are not accessible with JAWS.

2.5 Conclusion on the preanalysis

In the preanalysis, different areas have been investigated. It has been established that in Denmark are three categories used to classify blindness. These categories are socially blind, practically blind, and totally blind.

Along with the definition of blindness, computer interfaces for visual impaired people were also investigated. This revealed that there are primary two ways how the interaction between the visual impaired person and the computer takes place. If the visual impairment is low, a screen magnification software can be used, while a screen reader and an optional Braille keyboard is needed if the visual impairment is so severe that input cannot be given trough visual means.

When it comes to music production and performance with traditional instruments it seems that there are no problems where a solution has to be found. This is due to the availability of Braille notation and software that can convert common digital note formats into Braille notes.

In regards to digital music production there does not seem to be a good solution for totally blind

[10] `http://www.indigo2.dk/primus/`

people. The plugin CakeTalking for the the digital audio workstation Cakewalk Sonar let blind people use speech synthesis together with the software. This enables them to use the software but the experience seems confusing since only one element can be controlled and accessed at a time. This is also highlighted by the FBLIND framework described earlier. The interview with the blind IT consultant from IBOS also clarified that only node programs are used by Danish blind people. The development and testing of a tactile interface in coorporation with IBOS could also help them to gain new insights into what other solutions exist to the problems they are trying to help their blind clients with. The use of haptic- and force feedback interfaces like The Moose and The DAMPER seems as a promising way to make digital music production accessible for blind users. However based on the scope of this project it is chosen not to pursue this area of research further.

One area that seemed especially confusing is the task of making a drum sequence since a certain degree of overview would be beneficial, an overview that cannot be given with traditional dialogue driven interfaces like CakeTalking for Sonar. The lack of a an interface for making drum sequences made specifically for totally blind people and their special needs further emphasizes that the area of digital music production trough the use of a tactile interface could produce the most novel and beneficial research. The final problem statement is therefore stated as:

2.5.1 Final Problem Statement

How can a satisfying tangible interface with a sufficient degree of utility be created that enables blind people to create drum sequences?

In the analysis it will be researched what features of a drum machine is necessary to provide a sufficient degree of utility.

Usability Goals

As *satisfying* and *utility* are used in the final problem statement there meanings will be further defined. These words will also be used for guidelines when testing the final prototype, to see if it has been a success. Those words were chosen by the inspiration from the concept of usability. Usability covers areas as usefulness(utility), efficiency, effectiveness, learnability, satisfaction and accessibility [23]. Utility has been chosen because a product has to be useful in order to be used in the first place. If the product is effective, easy to learn, etc, but not useable it will simply not be used. Satisfaction has been chosen as it refers to the users perception, opinion and feelings of the product. Since the target group for the project is quite small it is likely that only subjective data from a few test participants can be gathered. Since satisfaction is subjective but also indicative of the overall usability of a product it is chosen to see if the product has this aspect.

Target Group

It is chosen to focus on totally blind people and practically blind people since they seem to have the largest problem. Socially blind people are likely to have so much acuity that they can use traditional MIDI controllers and navigate through the interface of DAWs with the use of screen magnification.

Since it can be hard to recruit blind musicians for testing purposes it is chosen to not exclusively focus on blind musicians only. However, musicians would be preferable since they can provide the most extensive feedback on the final prototype, based on their musical knowledge.

Delimitation

Although many tasks in the digital music production do not seem to be supported for blind people it is chosen to place focus on a smaller subset of these problems. Through research done in the preanalysis it is established that the largest problem is the creation of drum sequences, due to the lack of overview. Creating a tangible overview of a drum sequencer will therefore be the main focus of the project.

Success Criteria

In order to evaluate the overall success of the project and the learning process involved in conducting the project. Three goals have been formulated which the final prototype should fulfil. These are:

1. The blind user should be able to produce rhythm sequences without any assistance.
2. The usability of the product should be deemed sufficient by actual blind users.
3. Blind users should be satisfied with the use of the product.

Chapter 3

Analysis

In order to answer the final problem statement, further analysis of the project area is needed. Since the goal is to develop a drum sequencer it will be investigated what constitutes such a device. Rhythm theory is also described as this will has big influence on the design of the product. Since the target group are blind persons, the FBLIND Model, will also be covered in more depth to assess how applicable it is in the scope of this project.However, first of all theory regarding the design of tangible interfaces will be discussed. This is done so the knowledge gained in the rest of the analysis can be discussed how applicable it is in regards to the design of a tangible interface section. In order to guide this process it was chosen to follow the ISO 13407 human centered design lifecycle model. A summary of this model can be found in appendix B.

3.1 Theory of Tangible Interfaces

Since the problem outlined in the preanalysis is sought to be solved trough a tangible interface it is necessary to get a better understanding of how to design them. This will be done in this chapter.

3.1.1 Characteristics and Benefits of Tangible Interfaces

Although most interfaces can be considered to have a tangible component they are not normally categorized as being tangible. Traditional computer systems, remote controls and any other device that requires the user to do button presses have a degree of haptic feedback when input is provided by the user. However they often use a display, LED's or sounds to convey the current state of the system. What makes tangible user interfaces special is that the physical components of the interface both acts a input device as well as representation of what state the system is in. One definition of what constitutes a tangible user interface proposed by Ullmer contains five properties [30]:

Physically embodied
: The digital information or functionality of the system is embodied in some sort of physical form. Where a normal interface would use some sort of visual device to provide feedback to the user a tangible interface mostly uses the state of the physical objects to provide feedback to the user.

Physically representational
: Each of the various physical objects that make up the tangible interface represents some part of the underlying model that is being manipulated by the interface.

Physically manipulable
The user is able to manipulate the interface of the tangible interface in a physical manner. Often the artefacts that make up the interface are graspable, which means they can be picked up an rearranged by the user to provide new input to the system.

Spatially reconfigurable
The physical placement, translation, removal, and rotation of input artefacts are the primary way in which the user interacts with a system. Although mechanical parts such as levers, knobs and pushbuttons can be spatially reconfigurable they do not allow to be moved away from their physical position on the interface. If they do allow to be moved around, as for example is the case with a remote control, it does not act as a way to provide input to the system.

As it can be seen it there a strong emphasis in Ullmer's definition that a tangible interface consists of many small artefacts that can be grasped and spatially manipulated. The reason for this is that it was used to develop the constraint and token based approach to tangible user interfaces described later in this section.

The use of a tangible interface holds several benefits over traditional interfaces. It has long been known that cognition and memorization not only happens in the head but also in the physical world. Actions like counting on the fingers, taking notes or make drawing to a lecture and keeping a calendar are all examples of this. A study by Norman and Zhang showed that people perform better the more informations are represented externally by the system so they don't have to memorize it internally [33]. Tangible interfaces take advantage of this by representing information in a physical way. Based on their study Norman and Zhang also noted that the limitations created by physicality helps the user to determine which actions are possible to do with the interface and which are not.

Some disadvantages of tangible user interfaces mentioned by Ullmer are primarily that they can be more costly to develop since they often require non-standard components and sensors. Tangible interfaces also tends to be more specialized than traditional interfaces which also makes them less economic viable than traditional interfaces that can serve a variety of purposes.

In regards to HCI several theories and concepts are relevant to consider in the design of tangible user interfaces. These are described in the next sections.

3.1.2 Direct Manipulation

Direct Manipulation is an interface tradition that has been popular since the advent of graphical user interfaces. The main concept of direct manipulation is that it shall be easy for the user to see the result of an action by providing continuous feedback from the system. A widespread definition of Direct Manipulation contains three properties coined by Shneiderman[27]:

1. Continuous representation of the object of interest.
2. Physical actions or labelled button presses instead of complex syntax.
3. Rapid incremental reversible operations whose impact on the object of interest is immediately visible.

By continuous representation of the object of interest is meant that the part of the underlying software model being manipulated at one time should be represented to the user during the whole

length of the interaction with that part. This has traditionally be done trough a visual representation but with the advent of tangible interfaces can this representation also be tangible.
The second property means that the user is not required to do a long series of commands before an action takes place. Said in another way, one button press should trigger one action.
The last property dictates that if the user makes an action, its outcome should be communicated by the system instantaneously as well as the result of undoing the action. But how quick does this feedback needs to be? In regards to general causality the consensus seems to be that the delay should not exceed 50 ms [31][16].
If the concept of instantaneousness is upheld it facilitates good feedback since it is easier for the user to link the provided feedback to the specific action. This should also be clear if one considers Norman's seven stages of action [21]. When feedback from the system is given at every action the user can take, the user will be able to more quickly formulate a goal and correct the use of the system if it is handled in a wrong way. By having the actions being incremental also provides better feedback since the user can then compare the change in the state of the system with the number of times the action was performed.

According to Shneiderman there are many advantages of systems that support direct manipulation[27]:

1. Novices can learn basic functionality quickly, usually through a demonstration by a more experienced user.

2. Experts can work extremely rapidly to carry out a wide range of tasks.

3. Knowledgeable intermittent users can retain operational concepts, even defining new functions and features.

4. Error messages are rarely needed.

5. Users can see immediately if their actions are furthering their goals, and if not, they can simply change the direction of their activity.

Hutchins, Hollan and Norman expanded on this concept. They proposed that the main advantage of direct manipulation is that it creates the feeling of what they call directness [17]. They hypothesized that the cognitive load required to use a system is inversely proportional to the feeling of directness. They also split the term directness into two different traits, *distance* and *direct engagement*. With *distance* is meant the cognitive effort between the users goals and accomplishing the goals with the use of the system. If the gulf of execution and gulf of evaluation is bridged, the system will have a short *distance* from goal to accomplishment and the *distance* will therefore be short [17]. The other aspect, *direct engagement*, is the feeling users have when they directly manipulate the objects in the system or said in another words a sense of first personness. Where directness can be measured in how effective the system is, direct engagement is a qualitative feeling that the user experiences when the objects represented by the system is perceived as the objects the system seeks to model.

Hutchins, Hollan and Norman noted that there are areas where direct manipulation has shortcomings. It is not good at representing repetitive tasks since the user have to repeat the task the desired number of times. In this situation it would be better to provide the user with a way to automate the process trough some sort of symbolic representation. Environments that supports scripting of actions are an example of this. A more serious problem in their perspective is that direct manipulation mostly focuses on representing a task in a way that the users normally think about tasks in their domain. As they say:

"But if we restrict ourselves to only building an interface that allows us to do things we can already

do and to think in ways we already think, we will miss the most exciting potential of new technology: to provide new ways to think of and to interact with a domain."

Another problem is the trade off between directness and generality. An interface cannot possess both traits so it is up to the designer to judge which is most important. Also, if the interface is too perfect a representation of the domain and does not provide any additional feedback the user may believe that there is a problem in the way the interface communicates the state the system but where the problem actually lies in the task domain.
Lastly they note that direct manipulation versus conversational styles of interaction each have their own strengths and weaknesses. They points out that often a combination of the two styles will be the most beneficial . More recent research also supports these claims[11].

The MVC and the MCRit models

In traditional computer interfaces is the interaction with the user broken down into input and output. Most often the input is provided by using a keyboard or pointing device such as a mouse while the output is mostly provided by visuals on a monitor augmented by sounds. The Model-view-Controller model describes this clear separation between the input, output and underlying software model that is being shown to and manipulated by the user. This model can be seen on figure 3.1

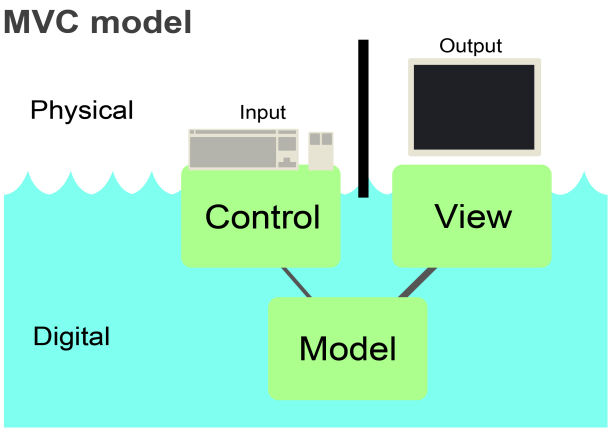


Figure 3.1: The Model View Controller Model

Based on the MVC model Ullmer has proposed a model that suits tangible interfaces. This model is called the MCRit model which stands for Model-Controller-Representations (intangible and tangible). This model can be seen on figure 3.2 .

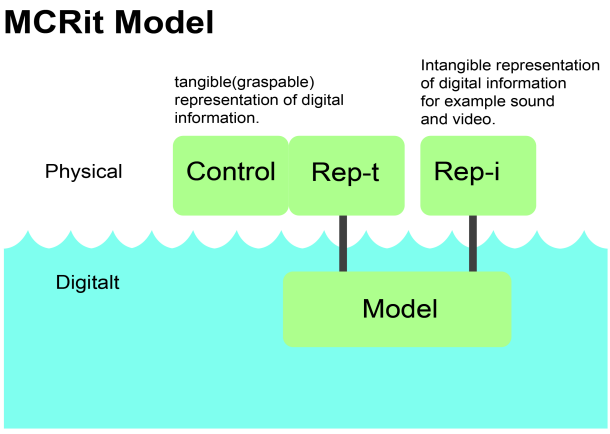


Figure 3.2: The Model Controller Representation(tangible/intangible) model

The MCRit model highlights how the physical objects used in a tangible user interface acts as a representation of the underlying model. This representation can both be manipulated and perceived by the user. The model also contains an intangible aspect of the representation which encompass visual and auditory feedback to the user. It is worth to note that the tangible aspect are often used to represent static parts of the underlying software model that only the user can change. The intangible aspect is thus often used to convey dynamic information that changes over time or as a result of changes made by the user.

3.1.3 Design Principle: The token+constraint approach

Early tangible interfaces tended to let the physical objects directly represent elements of a digital model that sought to represent an actual physical system. Physical objects where therefore mapped directly to their counterpart in the digital model. This meant there rarely where any abstraction in these interfaces. If the digital model for example was meant as a help to city planners the physical objects would be made to resemble houses, trees, and other objects found in a city. Since not all systems model a physical space the use of tangible user interfaces where limited.

Based on the four properties described earlier Ullmer developed the token-constraint approach. The idea is to use tokens, Ullmer's word for a physical object, to represent or be a token of aggregates of digital information. To make it clear to the user how these tokens can be interacted with they have to be placed in constraints which is simply containers, groves, or holes that the tokens fit in. This way the designer of a tangible user interface can build a semiotic for the use of the interface by restricting what tokens can be placed where. Constraints therefore acts as structures that physically channel how tokens can be manipulated, often limiting their movement to a single physical dimension inside the constraint. The process of placing a token inside a constraint is by Ullmer denoted association and the movement of the token inside the constraint is called manipulation. a token can also act as a constraint on it own and thus contain other tokens. This allows for grouping of information where Ullmer uses the concept of racks as an example. An overview of the token+constraint approach can be seen on figure 3.3

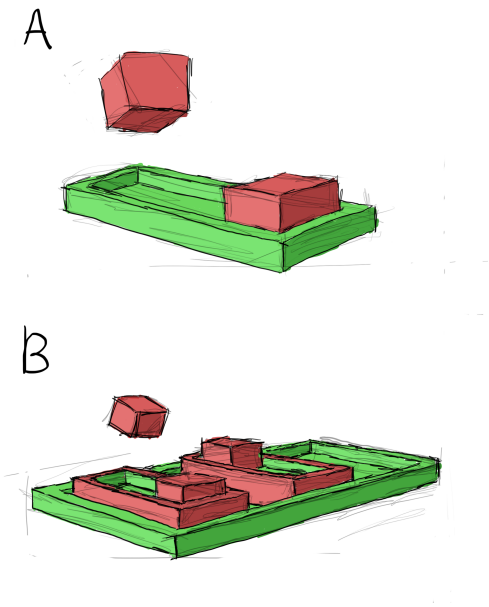


Figure 3.3: The token+constraint approach. Part A shows a single token(red cube) and a single constraint(green container). Part B shows how a constraint can also acts as a token in an even larger constraint.

The benefits of using a token+constraint approach according to Ullmer are:

1. The mechanical nature of the constraint can express digital/physical compatabilities between subsets of tokens. If two blocks have the same shape but different color they fit into the same constraint and can therefore be grouped.

2. The mechanical nature also means that tokens can only be placed in ways that are computationally sound so the input to the system is always valid.

3. The well defined borders created by a physical constraint makes it clear where one constraint ends and another begins.

4. Simplify perceptional load on the user by making it clear trough the use of physical constraints what can be done and not done in the system.

5. provides kinesthetic feedback due to passive haptic feedback from the interface.

6. If it is known that the user has to place the tokens in a constraint it makes it easier to make the system sense the presence of the blocks than if they where placed in a continuous space. It therefore becomes easier to implement the system.

Several mappings can be made between the digital model being manipulated by the interface and the relationship between tokens and constraints. The ones Ullmer proposed are in the table below:

Physical Relationships	Interaction Event	Digital Representations
Presence	Add/Remove	Logical Assertion, activation, binding
Position	Move	Geometric, Indexing, Modify Scalar
Sequence	Order, Change	Sequencing, Query Ordering
Proximity	Change in Proximity	Relationship strengh
- Connection	Connect/Disconnect	Logical Flow, Scope of influence
- Adjacency	Adjacent/Nonadjacent	Booleans, Axes: other paired relationships

Table 3.1: Mappings possible in the token+constraint approach

The BeatBearring described in the pre-analysis is a good example of a tangible interface where tokens and constraints are used. The metal balls acts as tokens that represents the individual sounds and the acrylic plate they are placed on acts as the constraint they are placed in.

3.1.4 Conclusion and Discussion on Theories for Tangible Interfaces

The token+constraint approach proposed by Ullmer's approach would have been useful if the project was not to design an interface for blind people. The main benefit of the token+constraint approach is that the interface's physical parts can be grasped, taken up from one constraint, and placed in another constraint. This has many advantages both on the cognitive and perceptional load on the user and on the implementation of the device. However the fact that the tokens can be removed from the interface would make it hard for the blind user to keep track on where tokens that are not placed in the interface are located. This would force a bigger cognitive load on the blind user since it would be necessary to remember where each token are located. The risk of moving a token by accident it also adds to the problems associated with the token+constraint approach in regard to this project's target group.

On the other hand is the tangible and intangible aspects encountered in the MCRit model worth to consider in relation to this project. Since a drum machine both contains static elements only manipulated by the user such as the sequence of sounds and dynamic elements such as the stepwise playing of the sounds it is clear that it is necessary to describe what elements are best represented in a tangible and intangible way.

The paradigm of direct manipulation also seems to be relevant as many of the benefits are good in the context of this project and not many of the disadvantages of the paradigm apply. For example is the task of creating drum loops a creative explorative process and therefore does not involve a lot of repetitive tasks to be done.

One disadvantage of direct manipulation has been that it leads to systems that does not support new ways of solving problems in the domain in question. However since this project tries to make an existing solution, the drum sequencer, accessible to blind users it is not a concern that the way to interact with drum patterns are seen before. Blind people cannot do this easily to start with so just being able to do it is a good goal in itself. A tangible interface like the ReacTable is also an example of a an interface that both supports direct manipulation and represents a task domain in a novel way[18]. This shows that it is possible to use direct manipulation to make novel interfaces.

Now where it has been assessed that a tangible user interface could indeed be useful to answer the problem statement it is chosen to look into haptic perception and gestalt laws to facilitate a

better design.

3.2 Haptic Perception and Gestalt Principles

Since the developed interface will rely on tactile feedback an overview of relevant aspects af tactile sensitivity, acuity and haptic perception will be given in this section. Unless stated otherwise is the information in this section and studies referenced taken from the book "Perception and Sensation" by Wolfe et al. [32]. First tactile sensivity and acuity will be elaborated upon.

3.2.1 tactile sensivity and acuity

In measuring tactile sensivity different tests have been used. One way to do this has been to see how much preassure was needed to be exerted on the skin before it could be registred. Max von Frey developed a method in where the thickness of a piece of string determined how much force it exerted on the test participant's skin. Another approach to measure tactile sensivity is to see find the smallest height of a raised elements on a flat surface one can detect. Trough testing with a criterion of 75% detection, that is 75% of the test participants had to be able to detect the stimuli, it has been determined that humans can detect a dot only 1 micrometer high.
When it comes to change in pressure over time, or simply tactile vibrations, a sensivity in a broad range of frequencies has been detected. Studies have shown that vibrations from 5 to 700 Hz can be detected.
Tactile acuity is normally measured as the two point threshold. This denotes the lowest distance between two tactile stimuli that can be detected as two points instead of one. This varies across the body but on the fingertips the two point threshold is 1 milimeter.

3.2.2 Haptic Perception - Identifying Objects

In haptic object recognition the texture gradient of the object are used more to recognize it than its percieved shape. The task of recognizing objects by touch is also something that we are quite good at. A study by Klatzky, Lederman, and Metzger from 1985 showed that test participants could recognize 100 common objects almost without error in 2 seconds on average.
Some aspects of haptic perception is subject to preattentive recognizion, that is that we are able to search for an aspect of the sensation without having formed a complete picture of what is percieved. This is known from the visual system where color recognition works in a similar way. A study found that it is easy to locate temperature, roughness, and hardness regardless if there was surrounding feedback to compare it to. These findings suggests that these aspects of haptic perception are preattentive recognizable. However not all haptic differences can be effectively searched for. The study also showed that it is hard to locate grooves oriented in a certain direction if other grooves with another direction are present.

The above study suggests that it is harder to recognize patterns than surface textures. However it is still possible to do so. Loomis has suggested that to some extent touch act sort of like a blurred vision when the fingertips are used to explore a raised pattern. It was tested how well people could identify different symbols including latin and Braille characters. In some test conditions the symbols was shown visually trough a blurring screen an at others they were presented as raised patterns. The study showed that the pattern of errors between the two conditions was the same, supporting Loomis' claim.

3.2.3 Haptic Perception - Localizing objects

In vision and audition the egocenter, the point that serves as the point of reference when determining spatial position of stimuli, is fixed. In haptic perception this is not the case. In haptic perception the egocenter change depending on which task we perform.

In order to have some more tools for the design of the interface it is chosen to look into Gestalt principles since these are often used in the development of existing types of graphical interfaces. The traditional gestalt principles seek to describe how the human visual perception groups objects together and find patterns in a scene. Much research has been conducted in this area and many different gestalt principles have been described.

3.2.4 Haptic Gestalt Principles

Research into Gestalt principles in regard to haptic perception is not as large as the body of research done on Gestalt laws regarding visual perception. However studies have been conducted that shows at least some of the Gestalt principles for visual perception are also applicable to the modality of touch. Chang, Nesbitt and Wilkins found that the gestalt principles of proximity and similarity apply to the haptic grouping of elements [7].

Quite a few Gestalt principles has yet to be proven experimentally to work trough the modality of touch. Since the gestalt principles are a powerful tool in the design of the interfaces it is chosen to look at other known gestalt principles in order to identify which ones could work with haptic feedback. This analysis is given below:

Good Continuation
: If two elements lie on the same contour they will be grouped together. This could hold true for haptic perception but would require that the elements are so close together that one by accident touches the start of the next element when the end of the current element is reached.

Connectedness
: If two elements are connected by something they are seen to belong to the same group. This principle is likely to hold true for haptic perception as well. As an example could a key chain be mentioned. Even though not all the things connected to a key chain is keys(law of similarity) they will still be grouped together because the keychain keeps them physically close.

Common Region
: If to elements are surrounded by another element they are seen as belonging to the same group. Again the keychain and its content is a good example that this is true. Here the pocket surrounds the keychain, supporting the feeling that the content of the keychain is belonging to the same group.

Common Fate
: Elements that move towards the same position will be seen as belonging to the same group. Unlikely that this will work with haptics since it requries an overview of all elements at the same time. If the notion of the elements is repeated multiple times it might work since it would then be possible to stick together the complete "picture" of the elements from individual touches. However since this Gestalt principle has a temporal dimension it would require that a person has multiple "pictures" of the seen to be able to piece together which elements move in which direction. All in all this principle seems to be hard to transfer to haptic feedback.

3.2.5 Conclusion on Haptic Perception

Several interesting findings can be concluded from this chapter. The two point threshold of 1 milimeter at the fingertips defines a lower threshold for how small parts of the interface can be. In this section it was also discovered that most humans are able to detect a change in height of 1 micrometer on a flat surface. This means that a rapid change in height larger than this is likely to be easily detectable.
In regards to object recognition, texture surfaces are easier to recognize than the shape of the object. This means that different textures are likely to provide a better way of grouping objects together than their shape.
The gestalt principles described and discussed in this section will also be valuable tools in the design of the final prototype.

3.3 The FBLIND Model

In the pre-analysis the FBLIND model was shortly described and had been proven to be very useful in the design of GUIs for blind people. As this report aims for making a tangible interface the models used to create the FBLIND will therefore be described step by step to see what influence the models when dealing with blind people, and what will be possible to use when making a tangible interface. user[1].

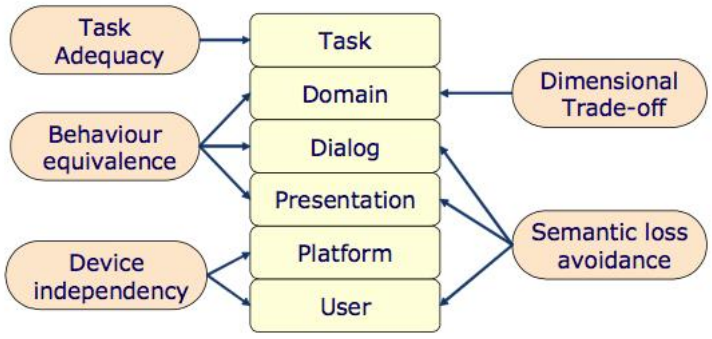


Figure 3.4: The FBLIND model

In the task model it is investigated what limitations blind people meet when using a specific HCI interface. As before mentioned a hand-eye coordination is not possible. The task model is used to specify the requirements for the interface.
The domain model is focused on the syntactic sequence of the interface objects and determines how a set of tasks is triggered. The main goal is to make the interface one-dimensional as there is a dimensional trade-off when developing for blind people. Normal sighted people can navigate on a screen with a mouse in a 2-D environment but when missing the vision, the hand can only move in one direction at a time, hence one dimension. It is therefore a good idea to present the interface hierarchically as a series of trees where the roots will be at the same place, so when navigating back you can easily find the other roots and start from there on a new tree. Though it is important to make the trees not to complicated because they do not have the ability to get an overview of where they are. When a change happen they should also be notified, for example if they change track on the drum sequencer. Traditionally this notification is given by voice or Braille.
Within the dialog model the commands and purposes of each function is defined. These commands

are triggered by interaction and can produce one or more system responses. It is important that the software has button-only access and if additional interaction techniques are needed, they should be incorporated so there will not be any semantic loss. This could be synthesized speech or Braille. When using the presentation model it is usually used as the model that describes the visual appearance of the user interface. Since this can not be the case for blind people it should therefore define the Braille and speech output for each user interface object, and also make sure that there is enough semantic information so it can be used as intended. It is also very important to be aware of the difference between using Braille and synthesized speech output. With Braille the user is in control of when the information will be obtained, which is not the case with synthesized speech whereas one has to wait for the speech to start and finish.
The platform model defines the capabilities, constraints and limitations of the target platform. Therefore the model should include, but not limited to, speech and Braille capabilities through standardized APIs such as [1]:

* For speech: text to speech output, stop speech output, automatic spelling and radio code spelling, management of queued speech messages, mechanism for the man- agement of speech progress, changing speech parameters

* For Braille: text to Braille output, providing information about the display characteristics (such as the display size), automatic management of text that is longer than the Braille display can accomodate, management of keys pressed in the Braille display, changing Braille parameters.

Lastly the user model is an overview of the characteristics of the stereotype user group. The model has to take into account configuration parameters due to possible semantic loss and also device-independency requirements e.g. speech parameters (speed, tone, volume, etc.) Braille parameters (Braille code type, presentation of the character attributes, etc.).

The FBLIND model concerns with making traditional graphical interfaces accessible to blind users, but the model does have elements that can be used when making a tangible user interface. The task model will also be required when making the physical interface, because one has to know what limitations there will be regarding a future product. This has already been done in the preanalysis. Another interesting aspect of the model is that it is also possible to make the tangible interface hierarchically, as described in the domain model. This could for instance be lines the blind people can follow with their hands. The dialog model requires a list of the different commands, in this case it will be what to use when making a sequence. Considering what the presentation model contains, it is highly relevant when making the interface. It is important to make sure that there will be no semantic loss. As suggested, Braille could be used, so when they find some Braille text on the interface it will give them information on where they are at the moment. Another way could be to add different textures to the interface. The user group is always important, no matter what product is being developed, and therefore also the model is relevant in the design of the product. This shows that the FBLIND model is very usable for carrying out the development of the final product, and will therefore serve as guideline.

The next section of the report will investigate some basic theory for creating drum rhythms to see what has to be considered when the product requirements for the design are formulated.

3.4 Rhythm Theory

A typical drum kit as the one illustrated in figure 3.5 contain at least one Kick, a Snare, Cymbals, and Tom-toms. The Kick and Snare are in general used to set the basic beat of the pattern. The

Hi-hat is crucial to include because it adds a realistic feel and rhythm to the overall sound. The remaining Cymbals are often used to accentuate climactic peaks and valleys in the music. Lastly the Tom-toms, or simply Toms, are most commonly used for fills. Fills are short sections of drumming that differ from the basic drum pattern to break up repetition and form semi-climactic peaks and valleys [3].

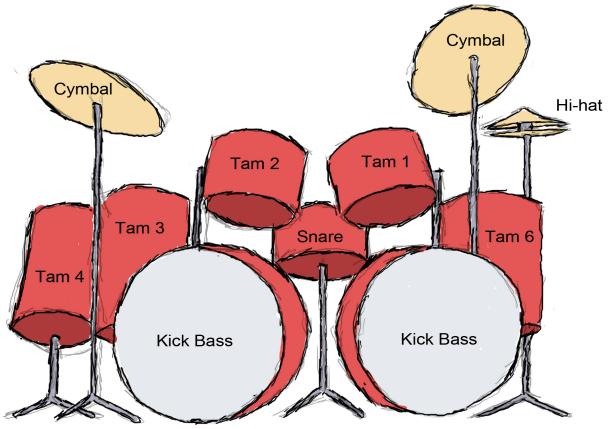


Figure 3.5: Traditional Drum Set

More relevant to this project are drum machines which's purpose is to facilitate the recording and playback of drum rhythms. A drum machine has a sequencer which can be programmed to trigger pre-recorded samples and the programming can depending on the machine be done in real-time or in step-time. Real-time programming is where the user plays the drum pattern he/she desires live while the machine is recording. More interesting for this project is step-time programming where the users place one sound at a time in the sequencer which as stated in the pre-analysis is a problem for visual-impaired people.

When using step-time programming the user has to recreate the rhythm and feel a drummer creates when playing live, the user therefore has to choose a tempo, time signature and quantization which suites the genre of music the user wants to create. The tempo is measured in BPM(beats per minutes) and most music genres are covered in the range of 50-250 BPM [3]. The time signature of the music is also important, because it decides the rhythm of the beat, and can be notated as a fraction for instance 3/8. In a time signature the denominator indicates the note value [1] which represents one beat and the numerator indicates how many beats there are per measure or bar. In the case of 4/4 *"there are 4 beats per measure with a quarter note carrying the value of one beat"*[3]. A measure or bar defines the length of a music unit so if the time signature is 3/4 and the tempo is 120 BPM the length of a bar is 1.5 sec. Since 120BPM/60sec=2 beats per sec. and one bar is (3/4)*2 sec.=1.5 sec. long. It can be a little hard to visualize the tempo and time signature concepts without actually listening to it so some examples of four bars at 120 BPM with different time signatures can be found on the DVD in the folder named "time signature". There is more to it than what is described above but an in-depth descriptions of note-theory and drum notation will be made at a latter point if it is necessary in order to complete this project.

[1] A note value indicates the relative duration of a note

Quantization is used to divide the measures into an equal number of pieces of time. 1/8 quantization will give the user 8 pieces and 1/16 quantization will give the user 16 pieces as illustrated in figure 3.6. The smaller the quantization is the more complex drum rhythms are able to be created.

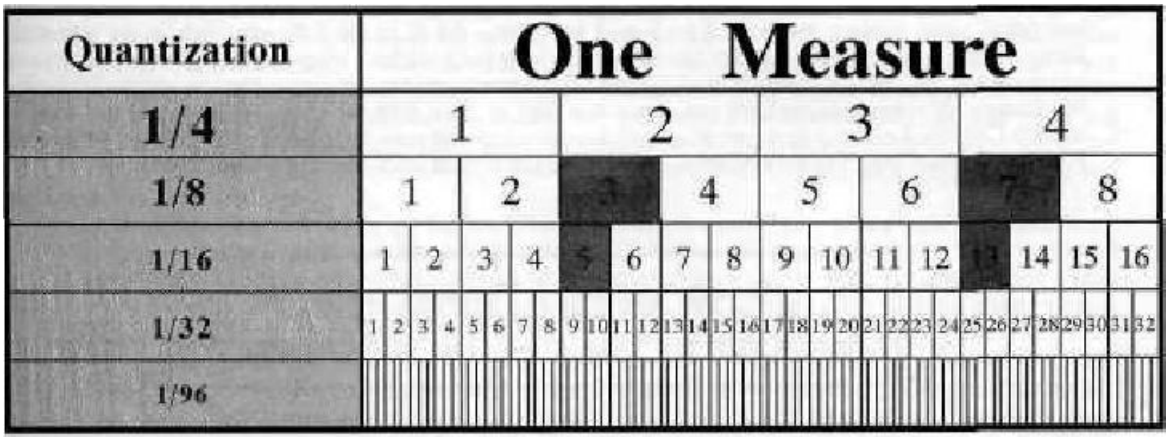


Figure 3.6: Quantization of a meassure[3]

In step-time programming the user can place a drum sound in each of these pieces. These pieces are then arranged in a lane for each type of drum that is present in the overall rhythm.

Conclusion on Rhythm Theory

These are the main areas which have to be considered when a drum sequencer is designed. It is therefore crucial for this project to consider which features to include in the design. All features are important for a musician but since the main focus of the project is to investigate how to design a step sequencer so blind people can control it, it should be considered to limit the features to the most important. Instead of trying to implement all the features like the opportunity to change time signature and quantization it could be advantageous in the first prototype to just implement one specific time signature and quantization to find out how to make that specific setup usable for a blind person. More features can then be added at a later point in the development. The chosen configuration could be bars with a 16/16 time signature and a 1/16 quantization. In the conclusion of the next section it will be elaborated upon why this is a good quantization to implement.

This section made it clear that most music genres are covered in the range of 50-250 BPM so that the user should be able to work within this range could also be a solution requirement.

The next section will go more in depth with music sequencers to give are more detailed understanding on how they work in practice which problems occurs when they are used by visual-impaired people.

3.5 Music Sequencers

Music sequencers are as mentioned in the previous section used for recording, editing and playing back music as digital audio and/or MIDI data. Most of today's music sequencers are software

based, for instance integrated in digital audio workstations, but they can also be found as an actual physical device. Related to the final problem statement, this section will mainly discuss hardware based sequencers, because of their tangibility.
Before the MIDI technology was introduced in 1983, music sequencers were operated by so called step sequencing. The new MIDI technology made it possible to play notes without restrictions, while step sequencing were based on playing sounds at a restricted pattern. Even though MIDI technology might be implemented in most modern sequencers this section will have a deeper look into step sequencing, because of its inherent constraints it is likely to be beneficial to solve the final problem statement [21]. Another reason, for sticking to step sequencing is, that most drum machines, like the Roland TR-808 Rhythm Composer launched in 1980 (see fig 3.9) are using step sequencing.
Whether implemented in software or hardware, these drum machines usually consist of 16 buttons/steps. Depending on the type of sequencer, these buttons are arranged in a line, as so called linear-style step sequencers, or in a circle, which are called circular-style step sequencers (see fig 3.7). Whether linear or circular each step/button can be switched on and off, and therefore a pattern will be created. The sequencer then converts the pattern into a drum-loop by going through the measure repeatedly.

(a) linear-style step sequencers (b) circular-style step sequencer

Figure 3.7: The "Mobius" linear-style step sequencer(a) vs. the "Orb" circular-style step sequencer(b), both sequencer are published by the company "Future Retro"

Because of general-purpose hardware samplers controlled by sequencers and software based sequencing and sampling, standalone drum machines are not common anymore. However, there are still some companies which are producing traditional drum machines, like Roland Corporation (under the name Boss).

It seems like the drum sequencers which have been investigated are designed with direct manipulation in mind as they inhabit the three properties stated in section3.1.2.In general the representation of the software model on these machine are visual. For instance is it common that light indicates which step gets triggered. It is also quite common that these machines only have one pattern which represents the drum sample of interest at a current time.
The buttons used to set the individual beats are often push buttons that do not give any tactile feedback about there current state. This state is instead conveyed trough visual means. This solution is not suited for blind people.
Several common features exist. These are control of the tempo, the opportunity to change drum sounds, solo and muting tracks and the possibility to play, stop and rewind the loop. They also

give the opportunity to work with more than one lane so you can create whole songs. The lane of buttons then represent one track at a time.
Besides these common functions others are more individual for the different machines. Some have different effects, others let you control the envelope of the sample and machines that use synthesized sound gives the opportunity to manipulate the sound in different ways. Most of the drum machines found during this investigation consisted of 16 steps where one step is corresponding to one beat of the time signature so with the 16/16 time signature and 1/16 quantization suggested in the previous section this would correspond to one bar.

Figure 3.8: The Boss DR-880

Figure 3.9: The Roland TR-808

Conclusion on Music Sequencers

From this section it was learned that music sequencers either are step based or real-time based and that they are either linear or circular in design. In the scope of this project it has been explained that step-sequencing is the best approach because of its inherent constraints.

Overall Layout

When it comes to the layout of the step sequencer there is the circular and linear design to choose from. Two aspects need to be considered, what is most easy to implement and what can work best

for a blind user. This will be decided in the design chapter.

Direct Manipulation

It seems like direct manipulation is good design choice when making a drum sequencer but the representations used on current drum sequencer are visual and the button gives no tactile feedback about their current state and are therefore a problem for blind users. This could be solved by using switches which would give tactile feedback about the state of the pattern but this requires a separate lane of for each type of drum to maintain the pattern.
It could be interesting to investigate how to make at least four tracks manageable for a blind user. The reason for this is that if there are three tracks the user would know that if there is a track above and below the current one, then he is on the track in the middle. With four tracks this possibility would be eliminated. If a solution is found that makes a blind user able to navigate between four tracks it could be a possibility that the solution would work on more tracks. Another reason why four tracks would be good is that the basic drums for making a drum loop as stated in the previous section will be covered.

Number of Steps

It seems like most drum machines uses 16 steps. If this was implemented it would allow for relatively complex drum patterns to be created. Another important reason to choose this number of steps is that with a lower number of steps, the blind user would not have to move his or her hands across the lane to interact with it. As a result less knowledge on how to design a tactile interface would be gained from the final prototype.

Other Common Features

There are also some common features that seems necessary to include in the design in order to test how these features can be presented for blind people in a tangible way. For example is the play/pause and stop buttons the most rudimentary feature since they control the playback of the drum sequence. Another common feature is to be able to mute lanes and soloing a track in order to help the user focus on one or more tracks. A way to control the tempo of the beat also seems relevant. The ability to select between multiple samples increases the virtuosity of what can be accomplished with the interface which is also worth considering to implement.

Finally it is necessary to convey what part of the sequence is currently being played. Based on the MCRit model described earlier it was discussed that parts of the representation that change are often best depicted in an intangible way. A panning of the created sequence or a sound could therefore be used to convey this information. As a result the next section of the analysis will be used to go deeper into how to correctly pan sound so it is conveyed to have a spatial movement by the listener.

3.6 Panning Sound

The primary way humans are locating a sound is by the difference there is in the time between the sound hits the two ears known as the interaural time difference (ITD) and in the difference of the sound level in the two ears which is called the interaural level difference (ILD) [32]. There are more processes involved when a human locates a sound such as the head-related transfer function but this function, but this is individual for everyone so it is not relevant to this project. ITD and ILD are on the other hand very relevant since these are physical information which can be simulated. When a sound comes from the left it will reach the left ear first and then the right. This difference is used to locate whether a sound comes from the left or the right. The azimuth is the term used to describe the horizontal location of a sound in an imaginary circle with the listeners head in the center [32]. The azimuth is measured in degrees with 0 degrees exactly in the front of the listeners and 180 degrees in the back. Figure 3.10 shows the ITD at difference degrees and it can be seen that there is no ITD when the sound is located directly in the front or back of the listener.

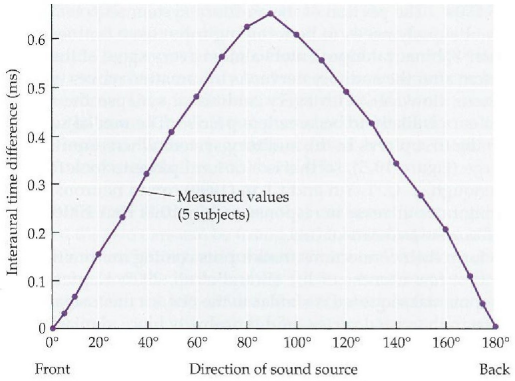


Figure 3.10: ITD's for sound sources varying from directly in the front of the listener to directly in the back [32]

The ILD works in the same way. When a sound comes from the left the level of the sound will be greater in the left ear than in the right and when a sound comes directly in the front or in the back of the listener there will be no difference. Figure 3.11 shows the ILD for tones of different frequencies presented in different positional around a listeners head and as it can be seen is the ILD's greatly reduced at lower frequencies.

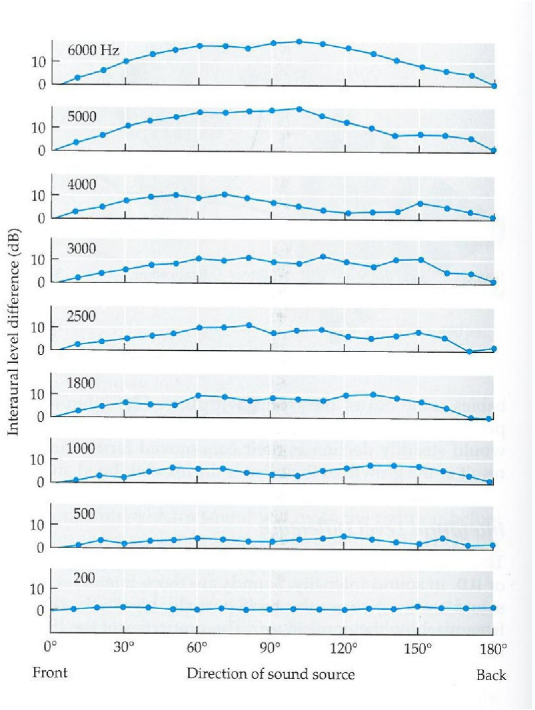


Figure 3.11: ILD's for tones with different frequencies at different positions around the listeners head [32]

The overall level of the sound, its frequency and the reverberant sound indicates the sounds distance to the listener[32].
Most music is recorded in stereophonic, commonly called stereo. This means that more than one audio channel and more than one speaker is used which then creates the illusion of directionality and audible perspective.The most commonly used stereo system when it comes to music is to have a separate channel for the left and right speaker. Sounds can then be panned to the left or the right speaker and at different sound levels to for simulate different locations for the sounds. This type of simple intensity panning only change the difference in the sound level and not the time difference. Other audio systems which uses more than two speakers is also available like for instance the 5.1 surround system which is used in most home cinemas. Such an audio system uses left and right speakers in front and behind the user and a speaker directly in front of the user to simulate the location of sounds by the use of stereo. Other more sophisticated ways of recording information about a soundfield and reproducing it is also available such as Ambisoncis. As this quote explains:

"Extensive listening tests over many years show Ambisonic recordings to be at least as good as any other form of recording at capturing sound images and far better than most, sound image." [2].

With Ambisonics you are not only able to record a soundfield but also to encode the location of a monophonic sound in all three dimensions along with the distance from the listener to the sound. It can then be decoded to give the best sound-image based on the amount of speakers used and their placement, a little section about the basic theory of Ambisonics can be found in appendix C.

[2] `http://www.york.ac.uk/inst/mustech/3d_audio/ambis2.htm`

Conclusion on Panning Sound

As mentioned in the previous section, a panning of the sequence or of a sound could be a way for a blind user to locate which step in the sequence is played. It has becomes clear that to implement this a sound system with multiply speakers are required to map the panned sound to map the steps in the sequencer. How many speakers that are necessary depends on the layout of sequencer but this section has stated that most music is mainly produced using a two speaker stereo system. Another information gained by this section is that the sound of the panning should not depend on low frequencies since the ILD's then will be reduced.

Since the project is targeted visual impaired people the next section will investigate if any precautions should be made when testing on impaired people.

3.7 Testing With Impaired People

One of the biggest problems when developing for impaired people is the availability of test participants. When testing on the general population the minimum of users to validate a test is around 20-30 people[19]. Though this can be difficult when the user are impaired, because it can be hard to find people with the specific impairment one is looking for, in one geographic area. Therefore the accepted approach when testing impaired people are small sample size, distributed research and in-depth case studies. When choosing the place for the test the best choice will be where the impaired person lives or is in everyday situations. This can be because they feel more comfortable being a place they know when they are testing a new product, and also because it can be hard for impaired people to use transportation. The downside of going to their place is that one does not have full control over the environment, and can therefore be hard to be specific when concerning about dependent variables. When it comes to research documentation, in this case for blind people, they are not able to read the document and instructions, so therefore this has to be considered in advance. One could for example read loud of a document and afterwards ask them if they accept the conditions, all this while recording on tape so their will not be any misunderstandings. When doing iterative testing to improve the overall interface it is often not possible to have impaired people testing all the time, as they can be hard to make schedules with. Therefore proxy users can be used. The bad thing about using proxy users is that they tend to be a lot different from the end user. Therefore when using proxy users, the test results have to be followed up by a test on real end users. The good thing about using proxy users is that the test can be conducted all the time, with not much of a time warning. Especially when on a university where there are people most of the time. With proxy users one tries to simulate the impairment of the end user. With blind people, the proxy user will be blindfolded to simulate a blind person.

3.8 Product Requirements

As a conclusion on the analysis, this section will describe the requirements the product needs to fulfil in order to satisfy the goals of the final problem statement.

Functional Requirements

Give an overview of the sequence created trough a tangible representation

By providing a tangible representation of the sequence it will hopefully be easy for the blind

user to gain an overview of the created sequence.

Use step based sequencing
Step based sequencing provides constraints to the interface which makes it easer to use than an interface where the placement of sounds is continuous. Step based sequencing is also easier to implement as a tangible interface.

Faciliate Direct Manipulation
Direct Manipulation interfaces has a long range of advantages over dialogue driven interfaces that blind users normally are forced to use. The task of creating sequences in a step based manner lends itself well to direct manipulation interface so the product will be developed to suit this type of interaction.

Convey what part of the sequence is currently played to the user
In order to provide good feedback to the user regarding the state of the system it is decided to use the panning of sound to convey this.

Have at least four separate tracks
Four tracks is required so the created sequence can contain the traditional percussion elements Kicks, Snares, Cymbals, and tom-toms at the same time.

Have the basic functions of a music sequencer
It is required that the user is able to play/pause, rewind, change samples and mute the individual tracks since these are common functions for music sequencers. This is done to ensure a sufficient degree of utility.

It is not considered to be important that the prototype can interface to commercial DAWs since this would not help answer the final problem statement. As a result the prototype is only required to be able to play the created rhythm and not send it as a MIDI signal to a DAW.

Data Requirements

Have the capability to sample a high number of inputs
Since the product will be based on 1/16 quantization and have four tracks it will need 16*4 = 64 Inputs to control the individual beats. This is required if the interface is to support Direct Manipulation of the rhythm sequence. On top of that is it likely that more inputs will be required to make the product functional like start and stop buttons.

Have a high sampling rate of input
Since the interface will be build upon the principle of Direct Manipulation the sampling of input should happen so fast that the system seems to register changes instantantiously.

Play sounds in standard definition
Since the main focus of this project is to make it easy for blind people to edit a rhythm it is not important that the sound quality of the product is high. That is something that can be improved upon in further iterations of the product.

Context of Use

Be able to withstand to be touched a lot
The built quality of the prototype should be solid since its gonna be touched a lot during the final test.

Chapter 4

Design

This chapter will describe the design of the final prototype based on the analysis and product requirements. The main focus will be on designing the physical interface since this is the focus of this project. The software part will be implemented to fit the physical interface. First the overall layout of the interface will be chosen and then a series of usability tests will be performed to find the best way to pan the sound, which is used to convey what part of the rhythm is being played. Afterwards a series of usability tests are performed since many aspects of the physical design are still unknown. These usability tests will give a solid foundation for a detailed and precise description of the final design. All the tests will be performed on proxy users, which will be mediology students. A chapter about usability testing and the guidelines used in the tests can be found in appendix D.

4.1 Linear versus Circular

One of the product requirements is to give the blind users the ability to identify which beats in the sequence are currently playing. This is important to keep in mind when choosing the layout of the sequencer. As mentioned in the end of the analysis this is sought to be done by panning the sound. This affects the overall design of the sequencer since the design that best facilitates this mapping should be chosen. Another thing that should be considered is the practicality of the design. On figure 4.1 are three different setups illustrated that shows how this mapping could be done.

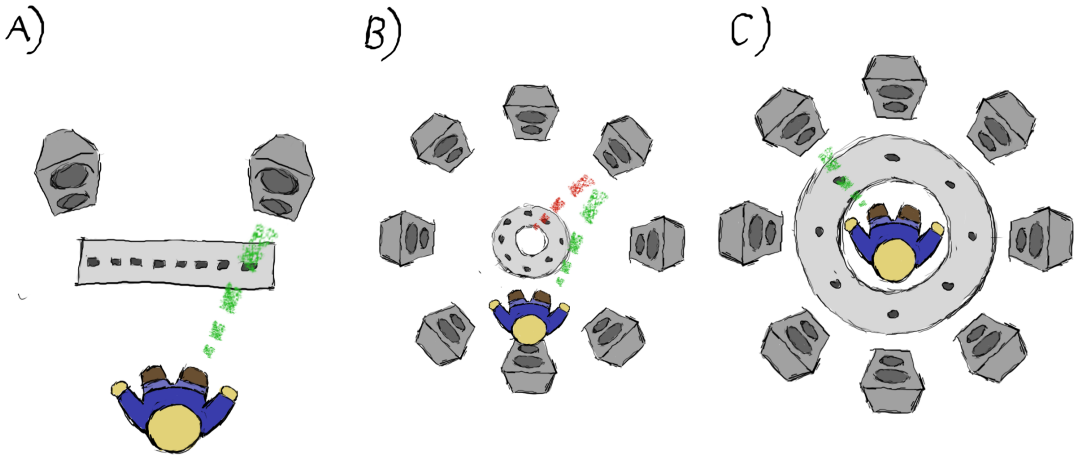


Figure 4.1: Different setups for panning. A is the linear setup which provides the best mapping between the panning and the current position in the sequence. Figure B and C are examples of how to use surround sound to pan the sound. with setup B the user has to project himself into the center of the sequencer. This cognitive load is sought to be removed in figure C by placing the user in the middle

With a linear sequencer and a two speaker sound setup the steps can be mapped as going from left to right by panning the sound from the left to the right speaker as illustrated in figure 4.1 A. In the circular design on figure 4.1 B the user would have to imagine that he or she is located at the center of the sequencer in order for the feedback to make sense. This would produce an unnecessary cognitive load whereas with the linear design the sound is directly coming from the direction where the current beat is currently located. In the other circular design on figure 4.1 C, the user is located in the center of the sequencer which could give a more natural mapping than the design in figure 4.1 B. However in this case the user would have to turn around in order to be able to reach all the steps.
It also has to be considered what is most practical and suited for producing music. This is a linear layout since it requires less space and a two speaker sound system is preferred for most music production. This would also give the user the possibility to use a pair of headphones. For these reasons the layout, in the final design of the prototype, will be linear and use a two speaker sound system.

The next section will cover the design of how to pan the sound.

4.2 Design of Sound Panning

In order to find the most precise way to pan the sound it is necessary to conduct tests of different panning methods. When the best method has been found it will be discussed which sound should be panned.

There will be made two tests which compare a step based and a continuous panning. One test will use a basic intensity panning with simple ITDs and a distance simulation[32],the other test will use Ambisonics sound processing to pan the sound. Both tests will measure how accurate a user can localize a step in a 16 step sequence and the two tests will be compared at the end.
The step based panning will be in groups of four steps, so for the first four steps the sound is panned all to the left and for the next four step it is panned closer to middle and so on. The two types of panning are implemented using MaxMsp 5.

Tests of Different Types of Panning

Each of these tests were executed on eight mediology students. The sequence started at a random place and stopped at a random place so it was not possible to count the steps to know which step was triggered without using the panning as a cue. The participants then had to guess at which step in the pattern the sequence was stopped by marking an X on a table of 16 columns and the test moderator then marked on a similar table where it actually was.

Below is the average difference between the actual place the sequence stopped and the guessed place for the intensity panning listed. The procedure and all results can be found in appendix D.1.

Results for test of intensity panning	
Average Distance from the Correct Answer - Step Based	1,6 steps
Average Distance from the Correct Answer - Continuous	2,3 steps

Table 4.1: Table of the average distance from the correct answer for step based and continuous intensity panning.

As seen in the average results the step based method is the most correct, and will therefore be used to compare the best results of the Ambisonics test which is listed below.

Results for test of Ambisonics panning	
Average Distance from the Correct Answer - Step Based	2,0 steps
Average Distance from the Correct Answer - Continuous	1,3 steps

Table 4.2: Table of the average distance from the correct answer for step based and continuous Ambisonics panning.

As seen in the Ambisonics test, the average of the continuous panning has better results than the step-based. Comparing the Ambisonics continuous panning to the step-based panning of the first test, the Ambisonics panning has the best result. It is therefore choosen for the final design.

These tests clearly indicate that the panning of a sound can give a pretty precise indication of which step is triggered in a 16 step sequence. The next design choice is which sound to pan.
If the sequence itself is panned the user will not know where he or she is if no steps are activated. On the other hand if a sound which is not part of the sequence, like metronome, is panned it will likely get very confusing when a lot of steps are activated. The choice lends to panning the sequence itself since the information provided from the panning only is relevant when there are activated steps in the sequence.

The next section will look into which sensors to use for controlling the sequencer and how they should be placed.

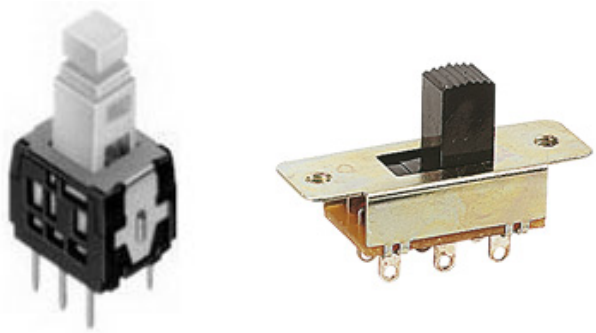
4.3 Sensors to Control the Sequencer and Their Placement

First part of this section will discuss which sensors will be used for the different functions on the interface and the next part will describe their placement. To follow the principle of Direct Manipulation all functions will be mapped to its own sensor.
When choosing sensors for controlling the interface it is important to keep in mind that the users are not able to get visual feedback about their state. A solution to this is to use sensors which gives the sufficient tactile feedback about their states according to the haptic perception described in section 4.2.

There are two fundamental difficulties when choosing and placing controls on an interface according to Norman[21]. The first one is the grouping problem or how to determine which switch belongs to which function. The other one is the mapping problem which can occur when there is a lot switches which will be the case on this interface. One solution to the grouping problem is to arrange switches that control one function apart from switches that control other functions. Another solution is to use different types of switches for different functions. These two solutions can also be combined. These solutions also match the Haptic Gestalt Principles about proximity and similarity described in section 4.2.3. Shape coding can also be used to distinguish switches with different functions[21] Based on the research into haptic perception described in section 4.2.2 shape coding might not be an effective choice for a tactile interface. Howeversince there has not been done any research in this area it will require additional testing.

When it comes to specific types of sensors, Rocker switches are normally used to turn something on and off and could therefore be used for muting the individual tracks, for turning the panning on and off and for turning the individual steps on the sequencer on and off. However according to Norman [21] it would be confusing to use the same switches for multiple functions.
A solution to this could be to use switches with different shapes or other switches like toggle switches and slide switches. By using toggle switches and slide switches another problem would occur since it can be hard to determine their state if you do not compare to a similar switch with the opposite state. These two types of switches would therefore not fit the panning or muting functions. However they could be a good solution for the steps since there are 16 in each lane so you could compare their states to each others.
Research has been conducted to see if choice of switch was affected by the tactile feedback it provided[20] but it could only be concluded that the availability of tactile feedback affected the choice of switches, not which ones were preferable. Therefore a test will be made that compares the two type of switches to decide which type to use for the steps.

Usability Testing of Switches



(a) Toggle switch (b) Slide switch

Figure 4.2: Picture of toggle and slide switch

Eight toggle switches and eight slide switches (see figure 4.2) was used for this test. The test was executed on six blindfolded mediology students. The first three participants tested the slide switches and then the toggle switches, the last three participants did it in reverse order to avoid the learning effect. For both types of switches the first four was set into a pattern that the participants had to guess using the tactile feedback they provided. They then had to recreate it with the last four. Time was taken for how long it took them to complete this task.
Below is shown a table with the average time used to guess and recreate the patterns. People were also asked which button they preferred after they had done the test. A full description of the test and all the results can be found in appendix E.1.

	Time before guessed the pre-made pattern	Time before recreated the pattern
Average Time Toggle Switches	26,9 seconds	19,6 seconds
Average Time Slide Switches	24,4 seconds	16,9 seconds

Table 4.3: Table of the average time used guessing a pre-made pattern and recreating the pattern for toggle and slide switches.

Five out of six participants had toggle switches as their favorite, two said that it felt more natural to push down, rather to push and pull.
When comparing the times recorded the participants were faster using the slider switches both when they had to guess the pre-made pattern and recreate it.
But since such a big a majority of the participants were most satisfied with the toggle switches they will be used in the final design for representing the steps.

Two different designs of rocker switches will be used for the panning function and for muting

the individual tracks. Wide ones will be used for the muting functions and one longer and more narrow one for the panning function.
For changing samples, rotary switches are a good solution since this function requires a switch that has more than two states. Each different sample will then correspond to one state of the switch and by using an asymmetric knob for the switch the user will have the opportunity to feel which state it is in. Slide switches with more than two states would also have been an option but would require more space than a rotary switch if a lot of samples were included.
For the tempo function a rotary potentiometer is a good solution since this variable change continuously. A slider potentiometer could also have been used but during the investigation of music sequencers it was observed that most used rotary potentiometers for changing tempo, moste likely because sliders are asociated with sound levels. The knob for the tempo should like the ones for the rotary switches also have an asymmetrical shape to provide tactile feedback about its state. Furthermore, it should have a different knob than the switches used for changing samples to distinguish the two functions from each other.
The last two functions are the play/pause and the rewind functions. Since both these functions control the playback of the loop they should perceived as a group. This will done by using ordinary push buttons with the same shape but different textures. The feedback from them will be the playback itself.

The sensors for controlling the sequencer is now chosen and the next part will discus and conclude on their placement on the physical interface. Two sections of controls has to be mapped, controls that only affects one track and controls that has a global effect such as tempo and playback. Therefore will all the global controls be placed in a section to the left of interface and the four tracks will be placed to right. By placing the global controls apart from the tracks it will hopefully be perceived as two different groups according to the gestalt principle of proximity and thus be easier to navigate.

According to the product requirements the sequencer should consist of at least four tracks. Since a linear layout was chosen the four tracks will be placed in four rows with 16 toggle switches in each(see figure 4.3 at the end of this section). This gives an easy and fast way for the users to get an overview of the pattern since corresponding steps in the four tracks are placed under and above each other. It will also give a two-dimensional representation of the whole sequence which could solve the mapping problem[21] mentioned previously. If the four tracks were placed after each other it would also make the sound panning used to convey what part of the rhythm is being played unusable.
The first track will represent the kick drum, the second track will represent the snare drum, the third track will represent the hihat and the fourth and last track will represent cymbal and percussion since this was the order in many of the drum machines investigated in the analysis. When investigating music sequencersit was discovered to be a common feature that the 16 steps were grouped into four groups, most likely to make the mapping of different time signatures easy (see figure 3.7a and 3.9). Even though the possibility to change time signature is not available in this prototype grouping the steps into sections of four will also be part of the design since it could give a better overview of which column the user is currently touching.
To get around the mapping problem, the muting switches and rotary switches for changing samples will be placed next to the track they control.
In the left section where the global controls are placed, the panning switch will be placed at the top. Below that will the tempo control be placed.
The last controls are the play/pause and rewind button which both control the playback of the sequence. They should be placed close to each other to signalize they control the the same thing and will be placed in the bottom of the section on the interface to make them easily accessible. Figure 4.3 below illustrates the design of the physical interface so far.

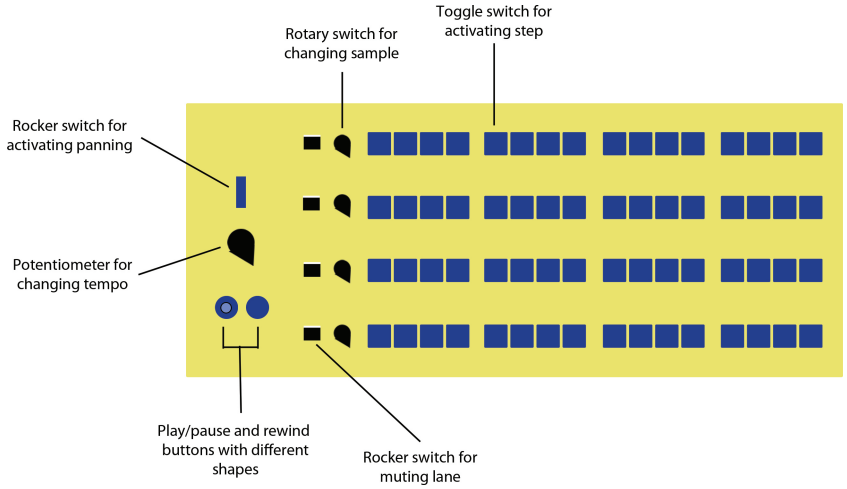


Figure 4.3: Illustration of how the sensors placement on the physical interface.

The last section will describe how the interface will be designed to make it easier for a blind user to navigate.

4.4 Step Switch Design and Separation of Different Sections of the Interface

During the analysis haptic perception and the FBLIND model was investigated and the knowledge gained will be used when designing the switches and to separate different part of the interface. A two part usability test will be used to make conclusion on this section.

The FBLIND model's domain model suggested that it is a good idea to design an interface for the blind as a three structure where one can follow a path back to the roots of the interface. This knowledge lead to the idea of using boundaries on the interface which the user could feel and follow to get from one function to another. Boundaries could also be used to emphasize group perception of the different tracks and separate the left and the right section of the interface according to the gestalt principle of common region.
When it comes to the toggle switches which represents the steps in the four tracks, a good balance should be found between grouping them together as on element of controls and as separate groups. The reason for this is that the have the same function just with different type of drum sound on each tracks. The problem is that with so many buttons it can be confusing for the user to find out which track they are currently touching just by placing each track in its own region within a main region for all the tracks. This is illustrated below in figure 4.4.

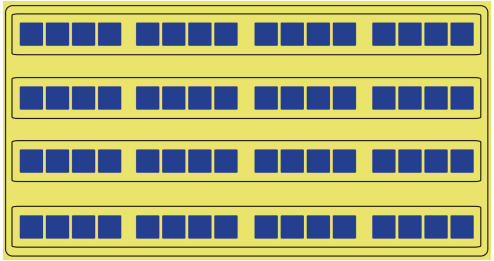


Figure 4.4: Illustration of one main region for the tracks and individual regions for each of tracks.

It is therefore important to find a good limit where the gestalt principles separate the tracks so strongly that the user perceives the different tracks as containing different functions.
Since it was discovered in the analysis that humans are good at recognizing different textures, the surface the switches are mounted on in each track will have different textures. If this is enough or if the switches in each track should have different shape in order to increase the perception of separation will be tested.
In order make the right design choice a two part usability test will be executed where the first part will test if the different tracks of the sequencer should be separated by boundaries. The second part will test whether the switches to control the steps should have different shapes or not. The test will be conducted on a low-fidelity prototype made of paper and clay.

Usability Testing of Boundaries and Different Shapes for the Step Switches

The test was executed on six blindfolded mediology students. In this test the participant had to locate switches trough the modality of touch. Two conditions were tested, one where the switches were encapsulated in boundaries and on were they were not. Tape was used as boundaries as it provided a texture that was very different from the texture of the paper the buttons were mounted on. Time was taken on how fast the participants were at finding pre-chosen switches. Due to the learning effect the first half of the participants starts with boundaries and the last of the participants starts with no boundaries. Each participant is asked to find the same switches so it is eliminated that it is easier to find some switches compared to others. They were also asked if they preferred with or without boundaries after the test. The average time based on the result from the test is shown in table 4.4.

Results for test of boundaries vs. no boundaries	
Average Time for all Switches with Boundaries	14,0 seconds
Average Time for all Switches without Boundaries	11,7 seconds

Table 4.4: Table of the average time for finding specific switches with and without boundaries.

The test procedure and all the results can be found in appendix E.2.

As can be seen there is a difference when comparing the average time for the two scenarios, 2,3 seconds. When looking at what people preferred the most, there was three on an interface with boundaries, two with no boundaries and one had no preference. It has therefore been chosen to use boundaries as there was a little majority on an interface with boundaries. Also one of the factors for using boundaries is the suggestion of making a tree model as previously described.
In the second part of the test it was investigated whether or not to use different shapes on the step switches. As with the test on boundaries, low fidelity prototypes made in clay were used. these are depicted on figure 4.5.

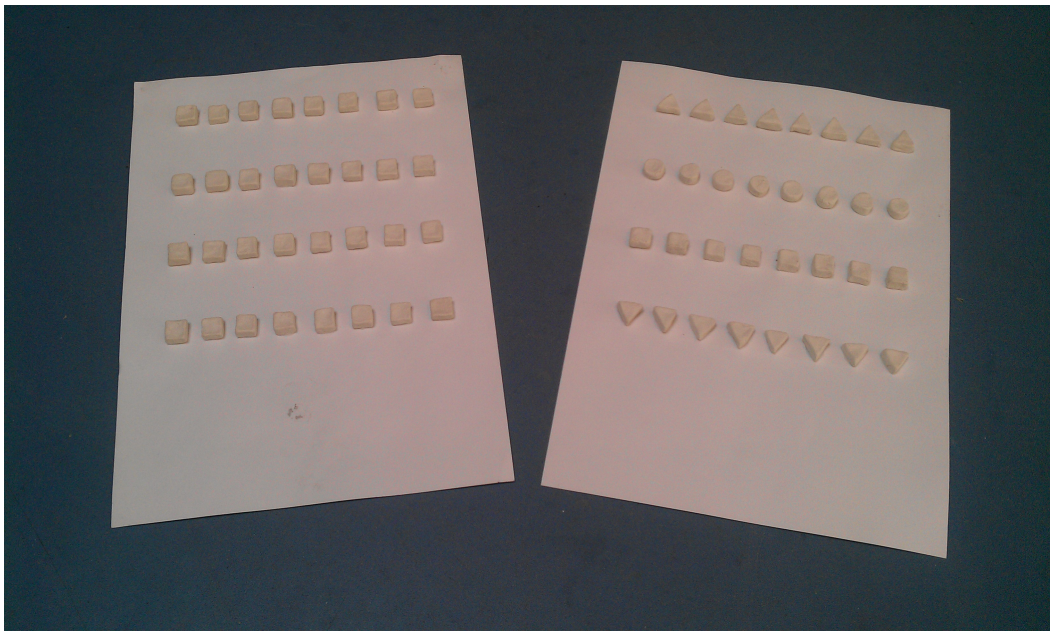


Figure 4.5: Picture of the the two prototypes used in the second part of the usability test.

as with the first part of the test, the participant had to find some pre-chosen switches by feeling with their hands on two different scenarios while they were blindfolded. In the first scenario the switches had different shapes and in the other they had the same shape. Again time was noted. As in the first part of the test they were also asked which of the two scenarios they preferred. Due to the learning curve, half of the participants started with one condition, the other half with the last condition. All the participants were asked to find the same switches, again to eliminate the possibility of a time difference in finding different switches. The shapes used can be seen in figure4.5. Below in table 4.5 is the average times displayed. All the results can be found in appendix E.2..

Results for test of different shapes vs. same shape	
Average Time for all Switches with Different Shapes	12,3 seconds
Average Time for all Switches with the Same Shapes	11,9 seconds

Table 4.5: Table of the average time for finding specific switches with the same and with different shapes.

The participants were in average 0,4 seconds faster when using the same shape. When asked what they preferred, four participants were on the switches with the same shape, one on different shapes and one did not have any preference. With this majority of participants choosing switches with same shape as their favorite, and with a little time advantage as well, it has been chosen to use switches with the same shape to represent the steps in the sequencer.

4.4.1 Conclusion on the design phase

Trough the usability tests performed in this chapter the final design of the physical interface has been established Figure 4.6 below illustrates the outcome of this process.

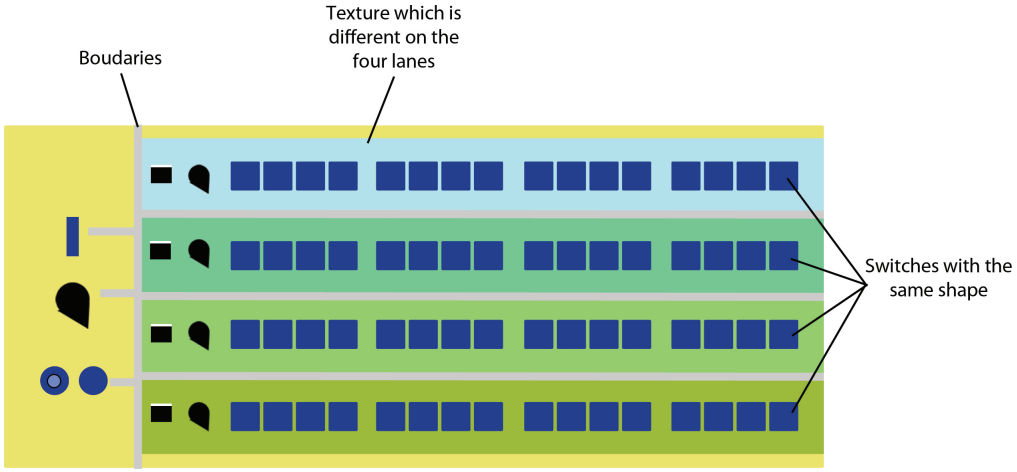


Figure 4.6: Illustration of the final design of the physical interface with boundaries and different textures on the four tracks.

The last product requirements are in regards to data and build quality which will be dealt with during the implementation.

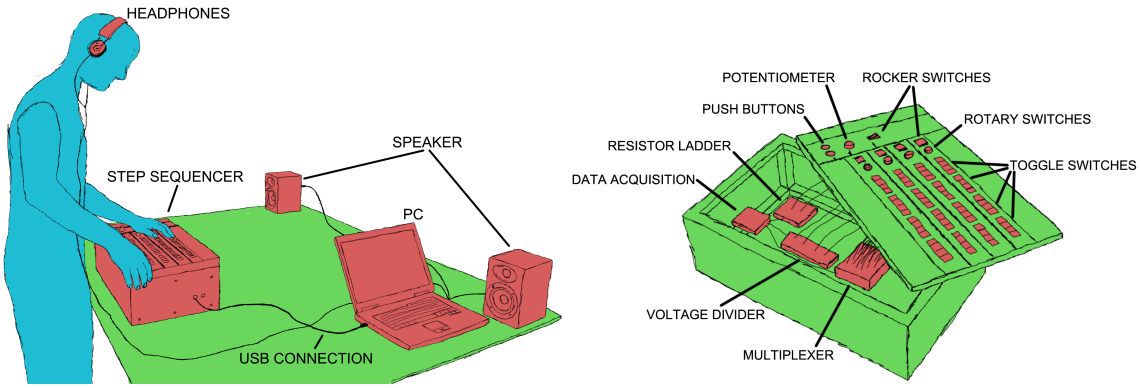
Chapter 5

Implementation

This chapter will explain how the design is implemented and is therefore divided into two section, where the first section looks into the implementation of the prototypes interaction mechanism using sensors technology and the second section explains how the software is implemented in Max/MSP.

5.1 Sensors Technology

Like discussed in the previous chapters of the report, a tangible step sequencer for blind people was built. This step sequencer, more precisely the data acquisition system, is connected through a USB cable to the PC, which runs the Max/MSP patch. The Max/MSP patch takes the input from the data acquisition system and generates the sound. A standard soundcard reproduces the audio and directs it to the headphones and/or speakers, which are worn by the user(headphones) and/or placed in front of the user (speakers) and provide him or her with auditory feedback. Figure 5.1a gives a overview off the total setup as well as a look inside the step sequencer.



(a) The total setup

(b) The step sequencer in detail

Figure 5.1: A sketch of the physical setup and a more detailed sketch from the step sequencer itself. It is just a sketch to illustrate the arrangement of the components, the amount of connections doesn't correspond to the final prototype.

The PC's screen seen in figure 5.1a doesn't provide any feedback for the blind user, it just helps to setup the application software by a normal sighted person. However, this section will focus on the implementation of the prototype's interaction mechanism using sensors, more detailed functionality of the application software can be found in the section Sequencer Software 5.2. Like seen in figure 5.1b the user can interact with the step sequencer by changing the input of five different types of sensors. Each type is used to solve different tasks, which is described in the design chapter. Figure 5.2 shows a photo of the final prototype, where the arrangement of the sensors can be seen. Further discussion about the sensors can be found in section 5.1.2 Sensor Electronics.

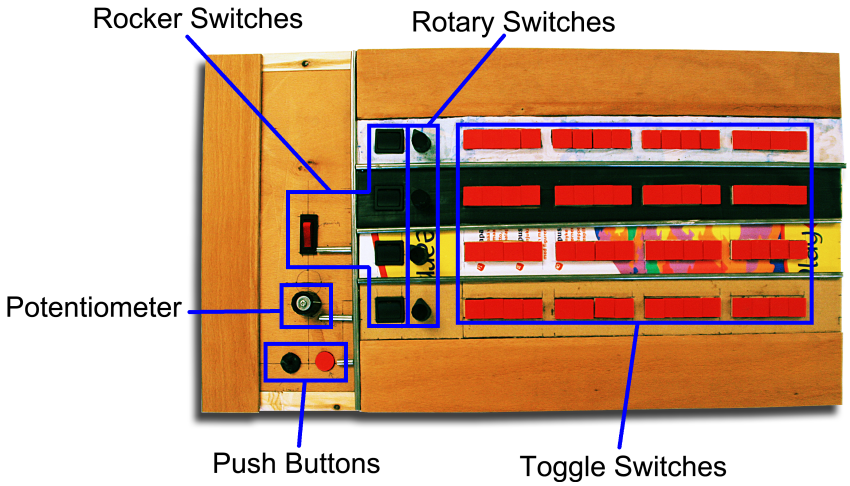


Figure 5.2: Photo of the final prototype and how the sensors are arranged

By adding these five types of sensors to the step sequencer interface, the user, by switching, pressing and turning the switches, buttons and potentiometer, will have control about the sounds the application software - implemented as a Max/MSP patch on the PC - will play. Due to the amount of sensors (four rotary switches, two push buttons, 64 toggle switches, one potentiometer and five rocker switches) implemented and the limitation of inputs at the data acquisition a multiplexer is needed, which converts several inputs into one output. Furthermore resistor ladder for the push buttons, as well as voltage divider for the rotary switches are added to reduce the amount of inputs to the data acquisition. More detailed description of the functionality and implementation of these sensors as well as the other circuits, including the multiplexer can be found in the section 5.1.2 Sensor Electronics as well as in the Appendix F. The technical details of the input interaction mechanism mentioned above and shown in figure 5.1, can be reproduced as a functional block diagram, shown below on Figure 5.3.

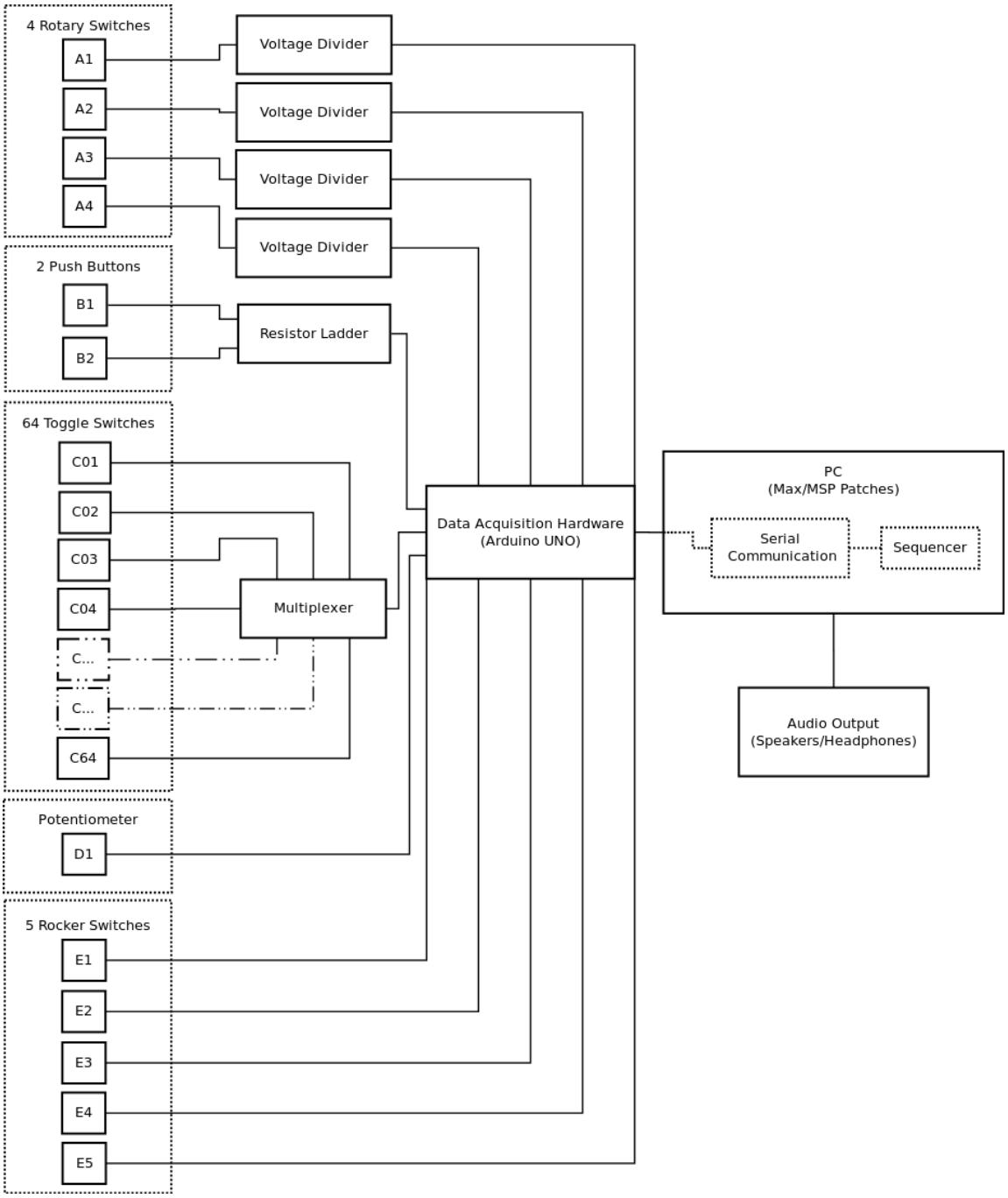


Figure 5.3: Functional block diagram of the input interaction mechanism

Concisely, the sensors produce analog voltage, depending on their physical setting. The data acquisition is able to receive analog voltage in a given range, and converts them from analog to digital. Where the data acquisition represents the analog voltage taken from the digital inputs as

LOW and HIGH and voltage taken from the analog inputs as a number. These digital values are then transferred to a PC, where a specially designed Max/MSP patch reprocesses them and makes them available for further use in Max/MSP. A brief overview of the most important issues are given in the following sections.

5.1.1 Data Acquisition

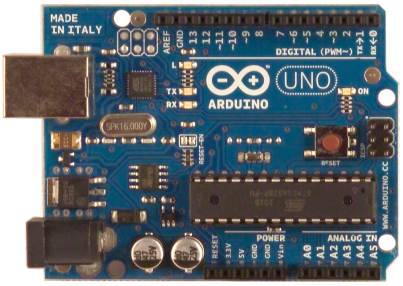


Figure 5.4: The Arduino Uno

To connect the tangible interface with the sequencer software it was necessary to use some sort of data acquisition device. This is done with the open source prototyping board Arduino Uno from the Italian firm Smart Projects depicted on figure 5.4. Besides being cheap and open-source the arduino family of boards also has an active community so it is easy to get help and find reference code and circuit designs. It is equipped with 6 analog input pins and 14 digital I/O pins[1].

The main part of the Arduino Uno is the ATmega328 microcontroller. Instead of having to program this with assembly code which is the normal way to program a microcontroller, code for the Arduino is made in the Arduino Programming Language which is essentially C with some constants and function calls specific for the Arduino environment. Where normal microcontrollers require a hardware programmer to put the program on the chip, the Arduino can be programmed by connecting it to a PC with a USB cable and using the Arduino software. This is possible because the ATmega microcontroller comes preloaded with a small program called a bootloader.

The arduino can either get its power trough the USB cable it is connected to the computer with or trough an external power supply. When the board is powered it can supply 5V to the circuit it is connected to. This means that it normally is not necessary to have an external power supply and just use the arduino connected to the PC to power the rest of the electronic components. Trough the USB cable it is able to have serial communication with the computer at a rate of up to 115200 baud[2] . Floating point values are sent as text, integer values as short ints (16 bits) and ASCII characters as chars(8 bit). The sampling resolution of of the arduino's analog to digital converter is 10 bits [2].

Arduino Program

Since this project requires the use of a multiplexer it was necessary to write a custom program for the arduino and make a MaxMSP patch that could communicate with this program. The same approach to serial communication called call-and-response or "handshaking" that was used in the

[1] `http://arduino.cc/en/Main/ArduinoBoardUno`
[2] Bits per second

provided programs was also used in the new program. Handshaking simply means that one of the parts in the communication keeps sending a known signal, usually an ASCII character, trough the serial connection. When this character is received by the other part it sends a value back based on the signal received.

The arduino program sends two integer values back when it receives an ASCII character from the PC. The first integer value is an ID that the MaxMSP patch uses to determine what sensor it is recieving data from. the second integer value is the actual sensor data. The arduino program only sends over a sensors data if there has been a change in the sensors' value since they were sampled last. In order to store the state of all sensors and initialize their value in the MaxMSP patch, all sensor data is sent over once when the patch is started. This is done by sending the ASCII character 's' to the arduino once for each connected sensor. When this has been done for all sensors, the MaxMSP patch begins to send the ASCII character 'u' over to signalize to the arduino that it should only send over values for sensors that have changed state. Since the values measured by the Arduino's DAC tends to be unstable, the program for the arduino only reacts to a change in the analog inputs if they exceed a certain threshold value defined in the program. Appendix G contains the code for the arduino.

5.1.2 Sensor Electronics

After gaining the necessary information about the data acquisition, the Arduino UNO, the following section will have a deeper look into the earlier mentioned circuits and sensors and how they are connected. Even though the following section will talk about digital states, it is important to know that all signals before entering the data acquisition are analog.

The core of the tangible step sequencer are the 64 toggle switches, where each is connected to the 5V power supply from the Arduino and to ground via a pull-down resistor. Based on their possible operating states (ON/OFF) they can be connected to an digital input of the Arduino. If a toggle switch's state is ON the digital input pin is connected to Vcc (5V), represented as HIGH. If they are switched OFF the digital input pin is connected to ground and therefore 0V, represented as LOW. To avoid floating between HIGH and LOW when the switch is OFF, a pull-down resistor was added. A more detailed description of toggle switches can be seen in Appendix F.1.

Due to the limitation of 14 digital inputs on the Arduino and the amount of 64 toggle switches a multiplexer was needed. This Multiplexer takes the 64 outputs from the 64 toggle switches and redirects them to a single output. This is done by several logic gates, which are controlled by the Arduino. Therefore the Arduino set which logic gate will be open and which input is let to the output. By going through all 64 inputs, one after another, and letting the signal to the output, before switching to the next input, all 64 inputs can be accessed. The switches signals itself (HIGH/LOW) aren't modified, but rather redirected and can therefore still be connected to a digital input of the Arduino. More detailed description of the multiplexer can be found in the Appendix F.5.

The five rocker switches have the same working principle as the toggle switches and therefore can also be connected to the digital inputs of the Arduino, where the state ON is represented as HIGH and OFF as LOW. Based on the small amount of rocker switches there is no multiplexer needed and they can be connected straight to the Arduino. More information about the rocker switches can be seen in Appendix F.1.

The four rotary switches can be seen as a switch, which connect one point to one of four other points. By turning the knob of the rotary switch it can be chosen which of the four points *1,2,3* or

4 will be connected to the point *A*. This principle can be treated like four toggle switches (connected to the same source), where always one of them will be ON and the other three OFF. As mentioned above, these ON/OFF states could be used to connect them to four digital inputs, where always one input would be HIGH and the other three would be LOW. These would require 16 more digital inputs (four rotary switches * four operating states). Due to the lack of digital inputs it was chosen to combine a voltage divider with the rotary switch. Each of the four possible states of a rotary switch will then be represented in a fixed analog voltage between 0 and 5V. By choosing adequate resistors the voltage difference between each state can be maximized, and therefore ambiguous states can be avoided. A deeper analysis of rotary switches and voltage divider can be found in Appendix F.2.

The two implemented push buttons are connected in a similar way as the earlier mentioned toggle switches. By connecting one pin of the button to a 5V power supply and the other pin to ground and a digital pin of the Arduino, it can be influenced by pressing the button if the digital input pin will be connected to 5V or not and therefore if it will be HIGH or LOW. Other than the toggle switches, which can be left in the ON state without operating them, the push buttons disconnect the circuit, except when they are pressed. Regarding there function, one of them play and pause the sequence and the other one rewinds the sequence, it can be excluded that they will be pressed at the same time. Based on this assumption the two push buttons can be connected to a resistor ladder, which converts there digital states (HIGH/LOW) to analog voltage. As the resistor ladder includes a voltage divider, the voltage output for each state can be adjusted by choosing convenient resistors. Further analysis of push buttons and resistor ladder will be handled in Appendix F.3.

The potentiometer for tempo control, can be seen as a variable voltage divider. Connecting the outer terminals to 5V power supply and ground, the wiper (middle terminal) gives the actual state of the potentiometer, represented as a analog voltage between 0V and 5V. Therefore it can be connected straight to one of the analog inputs. Appendix F.4 discusses the potentiometer more in detail.

This section gave a brief overview how the sensors and there related circuits were implemented. After connecting all sensors and circuits like described earlier, signals were obtained like expected.

5.1.3 Sensor Signal Software Preprocessing

When the sensor data has been retrieved from the arduino and read by the Max MSP patch that handles the serial communication it is sent to this patch to be processed into separate parallel signals sent to the sequencer patch. This process will be described in this section.

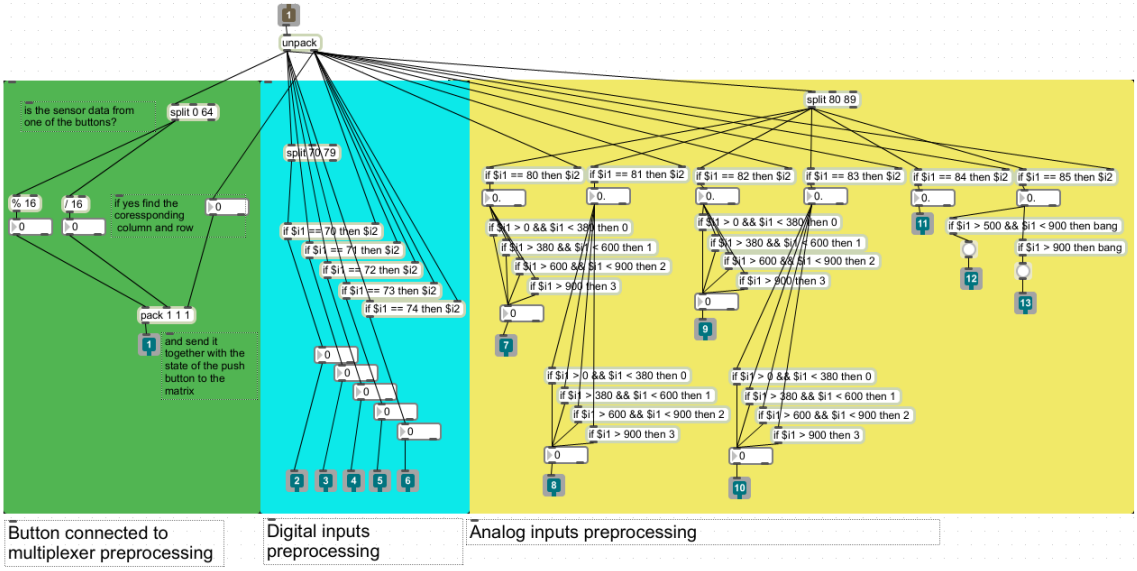


Figure 5.5: The sensor data preprocessing patch

As described earlier is the serial data received in the MaxMSP patch packed into a two item array, denoted a list in MaxMSP, and sent to the preprocessing patch. The first element is the ID of the sensor and the second is its value. The ID is used to determine how the sensor data is processed and where it is being sent. For all processing the list is split up into its two separate elements.

Sensor values in the range of 0 to 63 is taken as buttons and processed in the green part of the patch. First are all data packages with an ID above 63 discarded so only the buttons connected to the multiplexer is processed in this part of the patch. This is done with the Split object that only outputs values in its left outlet if they are in the defined range.The modulo and division of the sensor ID are then taken to get the column and row of the button. This is packed together with the state of the button into a list that thus contains the column, row and state of the button. This list is then exposed to the sequencer patch trough an outlet.
Sensors with ID's in the 70ties are buttons connected to the digital input pins on the Arduino and are processed in the turquoise part of the patch. Again a Split object is used to separate these data packages from the rest. If statements and the ID of the data package is then used to send it trough the correct outlet.
The last set of sensors, the ones connected to the analog input pins on the Arduino, has ID's in the 80ties. Again a Split object is used to only get data packages with these ID's The ID's from 80 to 83 are the rotary switches. Since these are implemented as a series of voltage dividers each state of the sensor produces a value from the Arduino's ADC. In order to map these values to a range of numbers that can be used in the sequencer patch a value for each state was first recorded down. Then ranges were defined where the recorded numbers lay approximately in the middle of their respective ranges. These ranges was then mapped to a digit from 0 to 3 trough the use of if-statements.
For the tempo control that has ID 84 it is simply sent directly to the sequencer patch.
The highest sensor value with ID 85 is the resistive ladder with the start/pause and stop buttons. Here the two button presses where separated by taking the value of each button being pressed and finding a value that is equally in between these two. Then if the value from the resistive ladder is

below this number it is the play/pause button and if it is above this value it is the stop button. This is accomplished trough two if-statements.

The next section will describe how the processed signals are used in the sequencer and how the sequencer is implemented.

5.2 Sequencer Software

The last part of the implementation was to program the music software which the physical interface controls. Figure5.6 is a screenshot of the main window of the MaxMSP patch which only consist of all the controls such as dial, toggle, and button objects and three sub-patches, where the programming of the different functions are encapsulated. There are five dial objects, one for controlling the tempo and four for changing samples on the individual tracks. Then there are six toggle objects, one each for playing and pausing the sequence, one for turning the panning function on and off and four for muting the individual tracks. The last control object is a button object which rewinds the sequence. Not all control objects are necessary but have implemented to give a graphical representation of what the user do on the physical interface. This makes it bugs in the communication between the physical interface and the software.

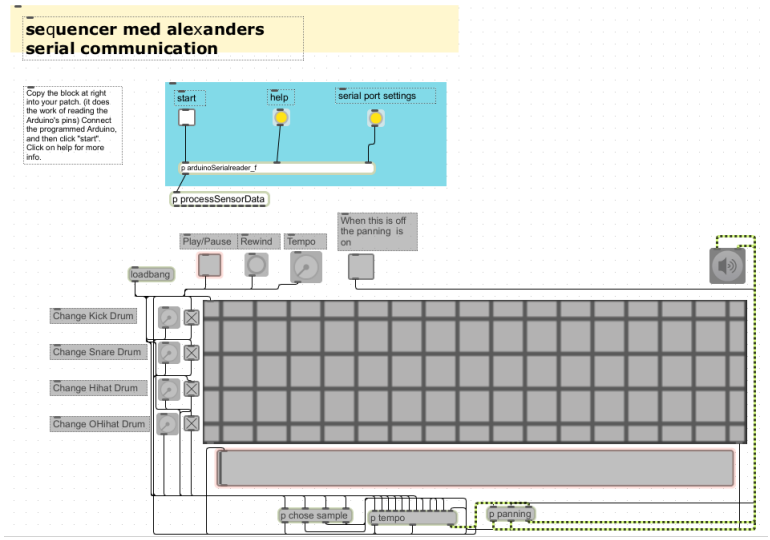


Figure 5.6: The Main Patch

A matrix object is used to store the pattern the user creates on the physical interface where each row represents a drum sample and each column represents a step in the sequence. When the play/pause button gets triggered, the metro object in the tempo patch (see figure5.7) start if it is paused and stop if it is playing. The metro object is connected to a counter object which counts from 0 to 15 with a speed that is dependent on the tempo of the metro object. This counter object is then connected to a message that recieves the column corresponding to the current count. When a column is selected, the matrix outputs the data from the column to an unpack object. This then triggers the four samples corresponding to the pattern created by the user. As can be seen on figure there is a gate object between the connections from the unpack object and each of the samples, these are connected to the mute buttons for each tracks so if a track is muted, its sample are not

triggered. The samples used are from a sample pack called Urban Warfare from *hiphiptools.com*[3].

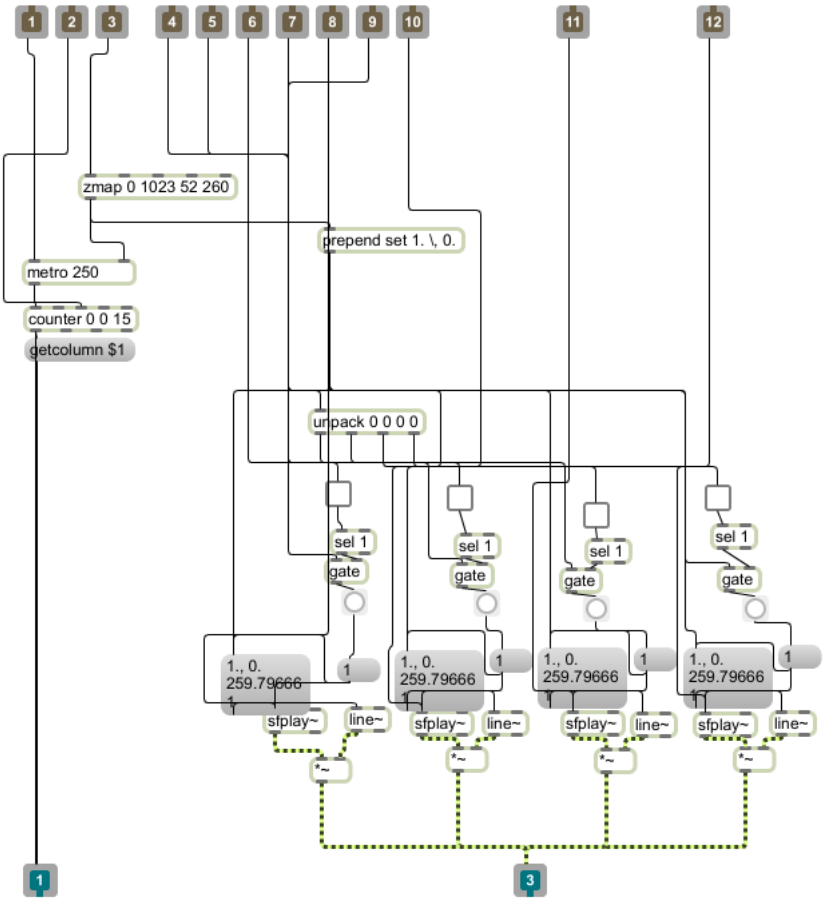


Figure 5.7: The Tempo Patch

The rewind button simply resets the counter object which makes the pattern start from the first column in the matrix. The last function in the tempo patch is to control the tempo of the loop. The output from the rotary potentiometer is in the range from 0 to 1023. This value is then mapped to BPM using the zmap object. This is connected to the metronome and thereby affect the tempo of the sequence. The mapped value also determines the length of the samples using a line object which output is multiplied with the signal from the sfplay object. This can be seen on figure 5.7. This was implemented because unwanted clicking sounds occurred if the tempo was so fast that the samples did not finish before it was triggered again. In a further prototype this function should be developed further so it only shortens a sample if another sample is triggered before the previous one ends.

[3]`http://www.hiphoptools.com/hip-hop-samples-hip-hop-sounds-drum-samples-urban-warfare.html`

There are four different samples of drum sounds for each track the user is able to switch between. This is implemented by naming the samples by type and a number for instance kick1, kick2 and kick3 and placing them in the same folder as the program. When the user turns the rotary switch to change sample the sensor preprocessing patch outputs a number between one and four. This gets combined with the type using a combine object in the chose sample patch which can be seen on figure5.8.

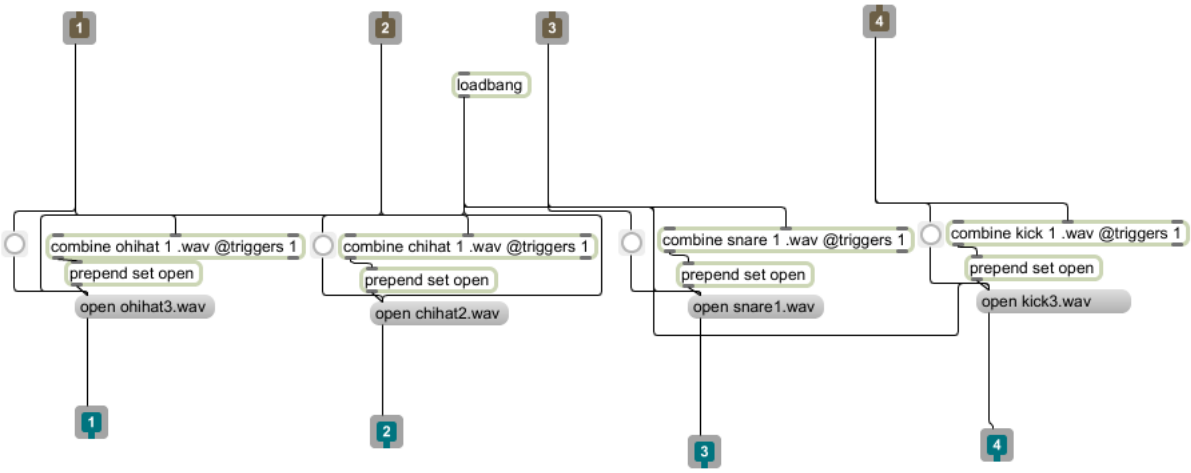


Figure 5.8: The Chose Sample Patch

The last function is the opportunity to pan the sequence to indicate which step is played. This is done with a toggle button object which switch a gate to ether output the original signal or to send it through an Ambisonics encode and decode object. The Ambisonics panning have been implemented using an external library made by Philippe Kocher and Jan Schacher which can be downloaded from the Institute for Computer Music and Sound Technology's webpage [4]. How this library has been implemented in MaxMsp can be found in [25] and [24]. The panning is made so at the first step the sound is all to the left of the azimuth and with a distance far away from the listener. Then as the steps get triggered the sound moves closer and more in front of the listener until it reaches the last step where the sound ends to the right on the azimuth with the same distance to the listener as for the first step. This is implemented using 3 scale objects and an if statement(see figure 5.9).

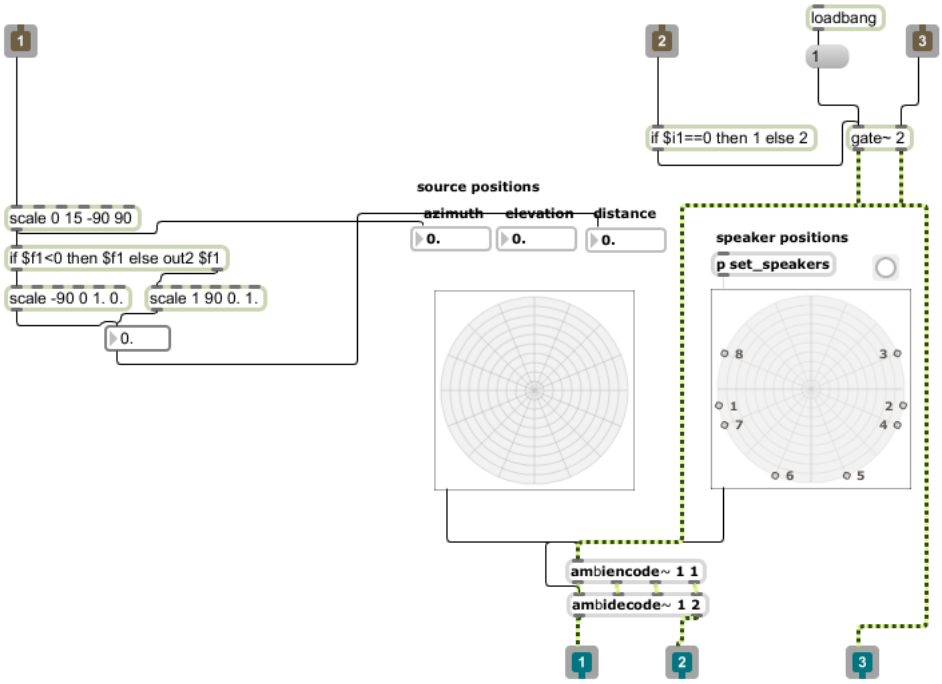


Figure 5.9: The Panning Patch

The first scale object maps the 16 steps in the counter from the tempo patch to -90 and 90 which is used to control where the sound is in relation to the azimuth. The if statement and the two other scale objects are used to control the distance from the listener.

This concludes the implementation chapter.

[4] http://www.icst.net/research/downloads/ambisonics-externals-for-maxmsp/

Chapter 6

Test

The test chapter will describe the test design used to assess the successfulness of the implemented prototype and the results of the test. The test design will describe what goal the test method aims to achieve, where it the test will be executed, and some changes that have been made to the method due to the test being conducted with a small sample. Afterwards results and observations will be listed.

6.1 Test Design

The final test will be formed with the goal of the summative usability test, which is to find out if the implemented prototype meets the product requirements. However since the test will be conducted with impaired people a few changes will be made to the method. Collecting quantitative data is not possible due to difficulties in recruiting enough participants needed for a true experimental test method.
Therefore quantitative data collection will be replaced with qualitative data, but the goal will remain the same. The participant will get as much time as they want to familiarize themselves with the interface and can ask all the questions they want. During this familiarizing the participant is encouraged to think aloud. After they think they have played enough around with the interface the test moderator will do an interview whereas the interface is commented upon. The interview will be focused on the usability goals defined in the pre-analysis, namely satisfaction and usefulness. The questions are semi-structured but open for discussion if new questions come in mind during the interview. A guide for the interview can be found in appendix H. The whole session is audio recorded on a Dictaphone. Furthermore are the participants' hands video recorded for further analysis. Both the audio and video can be found on the DVD[1].

To make it as comfortable as possible for the participants, the test is conducted at the Danish Institute for the Blind and Visually Impaired. Another reason for doing it the institute is to make sure that the blind people don't have to adapt to a new environment and which could lead them to feel insecure which would bias the test.

[1]Due to technical difficulties only two of the interviews are available as video

6.2 Test Results

For the test there were four participants. In the test results they will be referred as participant A, B, C, and D.
Participant A is educated in classical music. She has no experience with playing drums or in creating digital drum sequences. Participant B is a drummer and plays in a band, as participant A he has no experience in creating digital drum sequences.
The third participant, participant C, is a musician, whom has experience in creating digital music with help from his music teacher.
Participant D is the IT-consultant from IBOS that was interviewed in the pre-analysis. He has no experience in playing or creating digital music. Although he is the one knowing about existing programs for blind people, and is therefore aware of the programs used for creating digital music.
The test results will be listed in the different elements on the interface, and in the sections of the list, test results from A, B, C and D will be written.

General Observations

When observed the participants as they were familiarizing themselves with the interface, they all seemed to have an easy time learning the interface fairly quickly. When they played around with it, they did not seem to have forgotten anything as when the test moderator suggested them to use a function they found it without much help. None of them seemed to get angry or frustrated at any point. Participant A used around 11 minutes to familiarize herself with the interface, B used around eight minutes, C around 10 minutes and D played around with the interface for about 13 minutes. A and B did not feel on the box when the interview was executed but C and D did touch the interface while interviewing. Although A and B did not touch the interface they remembered the layout of the interface completely when asked into to the different controls.

Creating the sequence

According to participant A, the toggle switches were easy to handle although they did not seem to work all the time. A also had an easy time differentiate between the switches and could according to her own words:

> *"..not come up with any ideas for improvement".*

Participant B also had an easy time navigating through the switches:

> *"It is a good thing that you can feel that you just have pushed the button"*

Participant C said that it was very easy to navigate among them and easy to know where he was. He asked for a little more spacing between the buttons:

> *"if a final released was made there could be some more spacing between the buttons"*

Participant D said that they were easy to handle and that it was easy to feel the height difference between on and off. But he said that it was hard if they all were on or off to feel if they were on or off. He suggested rocker switches:

> *"...if it had been (rocker switches) instead"*

Braille on the switches would also make it easier to navigate around. He liked the response in the toggle switches.

Selecting Samples

Participant A liked the feedback from the rotary switches a lot. The clicking was a great indication that something had changed. She was also satisfied with the type of knob used. Participant B had more or less the same opinion as A for the rotary switches. He was very fond of the click feedback:

"it is a good thing that you feel something has happened"

Participant C likewise said:

"they are really brilliant because they gave such a good feedback by the clicking"

Participant D liked the tactile feedback they gave:

"you can also feel an arrow pointing"

Playback Controls

As for the start/pause and rewind buttons, participant A said it was good that different surfaces were used. Participant B said that the buttons were good, great functions. The same was said by participant C

"I think they worked fine, it is good that you can stop it".

Participant D was a bit more skeptical as he said during the interview

"it is a little hard to feel if you pause or start, , there should be a difference"

It should be noted that the play/pause and rewind buttons were not used that much under the process of familiarizing themselves with the interface.

Tempo Control

As for the potentiometer the three musicians A, B and C all used it right away to turn up the speed. Participant A said that she liked the knob for changing the speed, although it needed a tactile indication of its position. Participant B said he liked the knob used for the function:

"that was very good"

Participant C also like the knob and said it was very good, also a nice function. According to Participant D the knob only needed an indication of where it was,

"there is only missing an arrow"

Mute and Panning Controls

The mute buttons and the panning button were logical according to participant A, she compared them with light switches

"they are good, they are as light switches"

She therefore found it natural that it was switched up for turning them off and down for turning them on. The same was true for the switch used for the panning activation. As for participant B he said that the rocker switches was good, easy to use, and provided a great feedback. Participant C thought that the rocker switches was fine, it was good with one for each track. According to participant D the rocker switches were good. it was also very easy to feel what state they were in.

Boundaries and Surface Textures

As participant A commented on the boundaries used she said:

> *"I don't think they need to be there, I think it will be enough if there are a little spacing between the groups of buttons"*

Participant B also was of the opinion that the boundaries were redundant as he did not notice them. Participant C thought that the boundaries was good when he had to navigate around to the different areas of the interface. They were also good at splitting up the buttons, although he did not think they were necessary. With Participant D he liked the idea but said that it was redundant, as B.

Regarding the different textures used participant A found it unnecessary, and pointed out yet again that:

> *"you don't need that, no, I think the most important is that you can feel a difference between the buttons, you don't need to do more than that, as long as there is a good instruction".*

Participant B and D commented on the boundaries that they had the same opinion for textures as for the boundaries, they were redundant. Participant C said:

> *"it is (a) smart thing"*

but he did not notice it when he was familiarizing with the interface.

Panning Function

Participant A liked the panning very much as it gave her a good idea of where she was on the track:

> *"it comes from left to right and is the principle as the buttons,..., that is logical".*

When she was tested she would not really place an exact number, but guessed a little over the middle, and that was where it was. Participant B also liked the panning and said that it was a good idea but he would not use it himself as he counted the number of steps instead. The panning was therefore not tested with him. As for participant C he was very fond of the feature, and he said that it gave him a very good idea of where on the track he was, when he was tested he guessed two numbers from where it was. Participant D was tested to guess where on the track he was, and he guessed the exact number.

Overall Satisfaction and Usability

When it comes to whether or not that participant A she wanted to use the product, she had no idea, mainly because she was a classical musician and had no experience in drums. For participant B, he could use the product for practising when he was drumming, as something to play along with. As for participant C he said:

> *"I would probably use it, then I would avoid the troubles using JAWS, and getting that to work".*

Participant D did not think that he would be able to use for anything as it was not in his interest scope.
As for the whole interface participant A said that:

> *"the machine has to become a lot smaller so people would have space for it, ..., it would not do anything if the buttons became a lot smaller as well"*

Participant B was of the same opinion and had no problem if the interface became a lot smaller, he also asked for more authentic drum sounds. Participant A also said that regarding the design of the interface everything is possible as long as the instructions for the interface are good and precise, for example a CD with chapters of the different functions, buttons, etc. If possible there should also be less buttons, just as long as they were easy to differentiate. As when it comes to participant C he also pointed out that it would great if interface would become a lot smaller. He suggested that:

> *"then you just have some kind of switch so that you change between the tracks"*

Participant C had also tried CakeTalking for SONAR which he had experienced a demonstration of that he said:

> *"I would like to make music at home, but it is not what I want to use my computer for, it is so difficult"*

Later on, participant C also stated that,

> *"I see a potential in this one, , if it will become commercialized it will be probably be used by many"*

As for participant D he said that he had a hard time remember the functions, but participant A he said that it would be learnable with a good instruction. As the other three participants he also liked the idea of a smaller interface. He also stated that

> *"it is logical that the buttons are build up the same way in each track, , but it was hard to figure out where in the track the different buttons came in to play"*

Chapter 7

Discussion

In this chapter the results from the previous chapter will be discussed. The chapter will go through each sensor used for the final design by comparing the results from each participant, and try to evaluate on the given result. At the end will the overall satisfaction and utility of the product be assessed.

For the toggle switches used for activating the sound all the participants had an easy time navigating through them. They all liked the height difference there was between the two states of on and off, and the buttons also gave good feedback.
One of the participants asked for a little more spacing between the toggle switches so it would be easier to distinguish between them, and one had a hard time telling the state of the toggle switches when they were all on or off. This could indicate that if a future product was to be made, some changes would likely have to be made. For instance a spacing between the toggle buttons should be made and thereafter tested whether or not the other participants still had an easy time distinguishing between the toggle switches.
It would also be necessary to go further into the problem with the participant who had a hard time feeling the state of the buttons of they were all on or off. He suggested other buttons, and this highlights the problem with using proxy-users from the beginning. If the formative usability testing had been done on the possible end-users we could from there have chosen the proper switches to use. With using the end-users as a part of the formative process the results might have been different and therefore a whole other prototype would have been produced.

Looking at the knobs used on the rotary switches that were used for changing the sound, all the participants did very much agree that they gave really good feedback. It was also mentioned that the asymmetric form on the knobs gave a good indication of, in what state the rotary switch were. This could indicate that it was a success with knobs on the rotary switches, and also with the rotary switch itself as it gave good tactile feedback. The optimal would be to test on a lot of impaired people to see if they all liked the switch and knob, but as the scenario is at the moment this was certainly a success. Also with none of them had nothing on their mind of how to improve it. This also indicates that the rotary switch is a good switch to use, and therefore for future if could be considered to use a switch as this, just with another knob so that they would differentiable.

As for the play/pause and rewind push buttons three of them said that they worked fine, great functions. One of them said that it was easy to feel a difference between the two buttons as they had different textures, which worked fine. The last one said that he missed a difference between the states of play and pause. These buttons could be replaced with toggle switches and then be

tested to see wether or not people still would perceive them as working fine.

The rocker switches used for both the panning and for muting the tracks, all the participants agreed that they were great. They provided a good feedback and it was east to distinguish between the different states. One associated with light switches, which we had not thought off as they participants were blind. None of the participants had no improvements to think off during the interview. This could indicate that rocker switches are a switch which is a user-friendly switch regarding making a interface for blind people. This also indicates that rocker switches should be include in if a new prototype was to be made, yet again they should just have different size or/and shape if they goes different functions so they would be differentiable.

The speed function they all agreed that it was a good function. Three of them said the knob used for potentiometer was fine, the last one needed an indication of the state of the potentiometer. This would definitely be a great idea. Looking at the knob used for changing sound, it was stated that the asymmetric from of knob gave a good indication of, in which state it was. This of course should also be applied to the knob used for the potentiometer. This would most likely give the same positive feedback as it gave regarding the knob used for the rotary switch, but has to be tested before anything can be stated.

The boundaries and the textures used for the prototype was meant as guidelines and great distinguish opportunities when navigating through the interface, and thereby it should be difficult to get lost no matter where you were. Every participant thought that it was a nice idea although everyone also said that it was redundant. It was said that some people might use it, but in their case they did not need it. Since all participants did not use these feature it can be concluded that they were redundant. However this should absolutely be tested further before total exclusion from the final product. It would also be a good idea to test the interface without the boundaries and textures to see if the participants still would have an easy time navigating it. It was said that the buttons was enough to get a grip of where on were on the interface. This could indicate that we have underestimated the blind persons skill to get an overview as quickly as normal sighted people as normal sighted persons that have casually tried the product thought the boundaries and different textures were good.

As for the panning it was chosen to use the continuous ambisonic panning and all the participant thought it was a great function. Although one of them did not found it necessary as he had no problem listening to where he was. This feature is therefore likely to be a part of the final product as all the participants got a good indication of where on the track they were lwhen the panning was activated. It would be be good if it was tested on more blind people. Other possibilities for indicating where on the track the user is could also be considered.

In general everyone agreed that the box should be a lot smaller, and therefore this should be changed if a new prototype were to be made. It was stated that as long as there were very good instructions one way or the other, the interface could be as one wanted. Though it was still good with different shapes for the switches/buttons if they had different functions. One also stated that it would make the process much more likeable if this prototype was realised, instead of the already existing product for programming music. He referred to CakeTALK with Sonar, which was a very slow process. It can therefore be concluded that there is indeed a need for a drum sequencer and other MIDI controllers that can be used by the blind.
There should also be made very good instruction to follow the product. It was indicated that with a very good instruction everything seemed possible. The instructions could be made on a CD where there were chapters like chapter 1: buttons/switches, chapter 2: functions, etc.

Chapter 8

Conclusion

This project has sought to solve a problem the blind have with creating and composing music. Trough research it was established that digital music production likely held many problems for blind users. Various HCI theories were investigated and a tangible drum sequencer based on the principle of Direct Manipulation was constructed and tested with blind users. Based on the results of this test and the research done it can be concluded that there indeed is a need for a tangible drum sequencer and other interfaces that makes it possible for blind people to create and perform digital music.

In regards to the final prototype the blind test participants were in general wery satisfied with the tactile feedback provided by the interface allthough improvements could be made.The usability of the final prototype was also deemed to be sufficient.
Some commented that the toggle switches used to create the sequence were hard to feel the state of if all the adjacent ones also had the same state. A solution to this problem would be to align them perfectly with the surface they are mounted on when they are pressed down.
The concept of using a hiracical structure and grouping controls together with different surface textures was not used by the test participants in the final test. From this can be concluded that the target group's capabilities of haptic perception has been underestimated. If the final test had been performed with proxy users this valuable lesson would not have been learned.
The mapping of the part of the sequence being played to the panning of the sound was also a success. Several of the partcipants also noted that the interface should be smaller so it could more easily be transported around. The concept of having only one row of buttons that can be changed by the interface itself to represent more than one row was also suggested.

When seen in a bigger perspective it also seems this project has contributed something new in regards to HCI for the blind. Most interfaces for blind users use speech synthesis and braille output to communicate with the user which forces the interaction to be dialogue driven. The paradighm of Direct Manipulation has traditionally only been used to develop graphical interfaces where it has proven to be very effective. However as our final prototype demonstrates can this paradigm easily be adopted for blind users trough a tangible interface. It will be interesting to see if more interfaces for the blind in the future will follow the same trend.

Chapter 9

Future Perspectives

If the project is to be developed further, the lessons learned from the final summative test as well as new features will be incorporated into a new prototype. A focus group interview with the blind musicians that helped test the final prototype would also be conducted in order to further inform the design of a new prototype. A proposal for how this prototype could look like is depicted on figure 9.1:

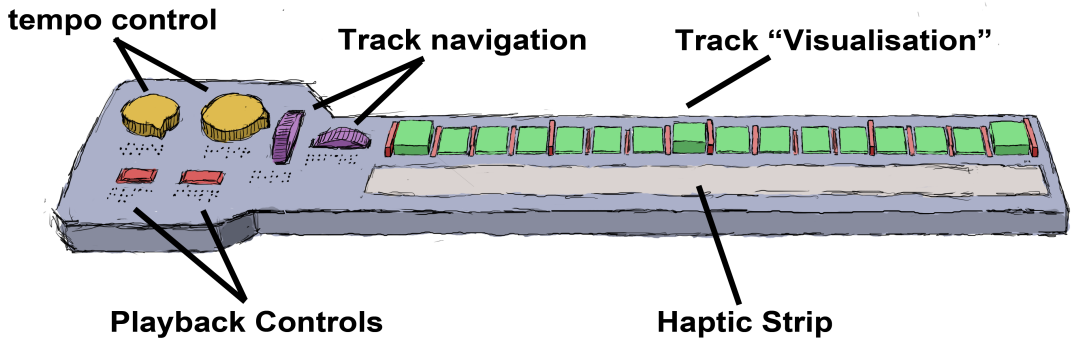


Figure 9.1: An example of how the next prototype could look like

The most notable difference between this proposal and the current prototype are that only a single lane of buttons is used. These buttons would be able to change state by the interface itself so the same lane of buttons could represent an infinite amount of lanes with infinite lengths. To navigate this matrix of beats two scroll wheels or maybe a joystick would be used. The change in height between the two button stages would also be bigger than it is for the current prototype to provide better tactile feedback. Furthermore the buttons would, when they are pressed down, be much closer to the surface they are mounted on as suggested in the conclusion.
To facilitate more than one time signature, barriers between the buttons, that are lowered and raised by the interface, would be able to mark different time signatures.
Even though the summative test proved that the panning of the sequence worked as intended it could be a nuisance in a concert or studio setting. One solution would be to use headphones and only play a panned version of the output from the sequencer in those. However it is likely that it is more desirable to keep the auditory channel completely free to other auditive inputs in the described settings. Instead could a haptic strip convey what part of the pattern is being played

by vibrating below the current beat. Instead of having one stepless knob to control the tempo, a two knob setup known from the classic drum sequencer Roland TR 808 could be utilized to make it easier to set the BPM precisely. Here a large knob is used to set the overall BPM and a smaller one is used to make smaller adjustments to it afterwards.
Braille labels would be used to mark all buttons and knobs to explain their function to the user. Finally speech output would be used to give additional information such as lane and beat number and tempo when these variables are changed by the user.

Although interfacing with a DAW and flexibility of the device was not investigated in this project, a successful MIDI controller based on the findings in this report would likely require a simple graphical interface akin to the one made for the tangible MIDI sequencer described in the pre-analysis. As such this interface would have to be easily read by a screen reader such as JAWS. This interface would allow the user to save and load patterns, change the samples being played, and generally perform task that requires direct interfacing with the DAW and the operating system.

If a device like the one described above was developed it would be beneficial to assess its usability. Since it has proven to be hard to recruit test persons for the tests and interviews done in this report, an extended use test is likely to be the best way to do this since it favors a high amount of data from a low amount of participants. In this extended use test, the participant would first get a copy of the product and an instruction in how to use it. The participant would then be encouraged to use the product over an extended period of time, likely a couple of weeks or a month. At the end would a follow up interview with the participant assess the usefulness of the interface.

Although the summative test concluded that the next iteration of the product should be smaller it has been observed by the project group that the relative large size of the product means it can be used in a collaborative way since more than one pair of hands can interact with it at the same time. This can be observed in the video "collaborative playing" located on the DVD accompanying this report. It should be noted that the persons in the video does not have a visual impairment. Further research is needed to see if the current form of the product also can facilitate collaborative work between blind users. If this is the case a version of the interface where the concept of having multiple lanes are kept could serve as a multi user version of the product. If this interface should be tested it could be done as a combination of a focus group interview and test session where the focus group together tried to use the interface.

Finally it should be noted that the project has attracted the attention of IBOS' music department and is likely to be presented at a symposium on music technology for the blind in the fall of 2011.

Bibliography

[1] F. Alonso, J. L. Fuertes, A. L. González, and L. Martínez, *User-interface modelling for blind users*, in Proceedings of the 11th international conference on Computers Helping People with Special Needs, ICCHP '08, Berlin, Heidelberg, 2008, Springer-Verlag, pp. 789–796.

[2] Atmel Corporation, *8-bit AVR Microcontroller with 4/8/16/32K Bytes In-System programmable Flash*, Atmel Corporation, 8271csavr08/10 ed.

[3] R. F. Badness, *Drum Programming: A Complete Guide to Program and Think Like A Drummer*, Hal Leonard Corp, November 1991.

[4] S. Bengtsson, N. C. Mateu, and A. Hst, *Blinde og strkt svagtsynede*, (2010).

[5] P. Bennett and S. O'Modhrain, *The beatbearing: a tangible rhythm sequencer*, in Proceedings of NordiCHI 2008: 5th Nordic Conference on Computer-Human Interaction (electronic proceedings), 2008.

[6] P. Bennett, N. Ward, S. O'Modhrain, and P. Rebelo, *Damper: a platform for effortful interface development*, in Proceedings of the 7th international conference on New interfaces for musical expression, NIME '07, New York, NY, USA, 2007, ACM, pp. 273–276.

[7] D. Chang, K. V. Nesbitt, and K. Wilkins, *The gestalt principles of similarity and proximity apply to both the haptic and visual grouping of elements*, in Proceedings of the eight Australasian conference on User interface - Volume 64, AUIC '07, Darlinghurst, Australia, Australia, 2007, Australian Computer Society, Inc., pp. 79–86.

[8] L. L. Chu, *User performance and haptic design issues for a force-feedback sound editing interface*, in CHI '02 extended abstracts on Human factors in computing systems, CHI EA '02, New York, NY, USA, 2002, ACM, pp. 544–545.

[9] L. Cocchiarella, A. M. Association, and G. Andersson, *Guides to the evaluation of permanent impairment*, Guides to the Evaluation of Permanent Impairment, American Medical Association, 2001.

[10] G. Essl and S. O'modhrain, *An enactive approach to the design of new tangible musical instruments*, Org. Sound, 11 (2006), pp. 285–296.

[11] D. M. Frohlich, *Direct manipulation and other lessons*, in Handbook of human computer interaction (2nd edition), Elsevier, 1997, pp. 463–488.

[12] B. Gillespie and S. O'Modhrain, *The moose: A haptic user interface for blind persons with application to the digital sound studio*, (1995).

[13] F. Gougoux, F. Lepore, M. Lassonde, P. Voss, R. J. Zatorre, and P. Belin, *Neuropsychology: Pitch discrimination in the early blind*, Nature, 430 (2004), p. 309.

[14] T. Haenselmann, H. Lemelson, K. Adam, and W. Effelsberg, *A tangible midi sequencer for visually impaired people*, in Proceedings of the 17th ACM international conference on Multimedia, MM '09, New York, NY, USA, 2009, ACM, pp. 993–994.

[15] J. J. Hofmann, *Ambisonics tutorial.* `http://www.sonicarchitecture.de/pdf/AmbiTutorial_en.pdf`, *Last accessed: May 2011.*

[16] D. H. Human and D. Heckenberg, *Using mac os x for real-time image processing*, in Proceedings of the Apple University Consortium Conference, 2003.

[17] E. L. Hutchins, J. D. Hollan, and D. A. Norman, *Direct manipulation interfaces*, Hum.-Comput. Interact., 1 (1985), pp. 311–338.

[18] S. Jordà, G. Geiger, M. Alonso, and M. Kaltenbrunner, *The reactable: exploring the synergy between live music performance and tabletop tangible interfaces*, in Proceedings of the 1st international conference on Tangible and embedded interaction, TEI '07, New York, NY, USA, 2007, ACM, pp. 139–146.

[19] J. Lazar, J. H. Feng, and H. Hochheiser, *Research Methods in Human-Computer Interaction*, Wiley Publishing, 2010.

[20] D. H. Mortensen, S. Bech, D. R. Begault, and B. D. Adelstein, *The relative importance of visual, auditory, and haptic information for the user's experience of mechanical switches*, PERCEPTION, 38 (2009), pp. 1560–1571.

[21] D. A. Norman, *The Design of Everyday Things*, Basic Books, New York, reprint paperback ed., 2002.

[22] J. Preece, Y. Rogers, and H. Sharp, eds., *Interaction Design: Beyond Human-Computer Interaction*, John Wiley, January 2002.

[23] J. Rubin and D. Chisnell, *Handbook of usability testing: how to plan, design, and conduct effective tests*, Safari Books Online, Wiley Pub., 2008.

[24] J. C. Schacher, *Seven years of icst ambisonics tools for maxmsp - a brief report.*

[25] J. C. Schacher and P. Kocher, *Ambisonics spatialization tools for max/msp*, New Orleans, US, 2006.

[26] P. Scherz, *Practical Electronics for Inventors*, McGraw-Hill, Inc., New York, NY, USA, 2 ed., 2007.

[27] B. Shneiderman, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st ed., 1987.

[28] J. Strong, *Home Recording For Musicians For Dummies (For Dummies (Computer/Tech))*, For Dummies, 2005.

[29] P. Szekely, P. Luo, and R. Neches, *Beyond interface builders: model-based interface tools*, in Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems, CHI '93, New York, NY, USA, 1993, ACM, pp. 383–390.

[30] B. A. Ullmer, *Tangible interfaces for manipulating aggregates of digital information*, PhD thesis, 2002. AAI0804673.

[31] C. von Hardenberg and F. Bérard, *Bare-hand human-computer interaction*, in Proceedings of the 2001 workshop on Perceptive user interfaces, PUI '01, New York, NY, USA, 2001, ACM, pp. 1–8.

[32] J. M. Wolfe, K. R. Kluender, and D. M. Levi, *Sensation and Perception*, Sinauer Associates, Inc., 2.ed ed., October 2008.

[33] J. Zhang and D. A. Norman, *Representations in distributed cognitive tasks*, Cognitive Science, 18 (1994), pp. 87–122.

Appendix A

Interview Guide

Semi-structured interview:

1. What is your experience with working with music and sound?

If the interviewed has experience with instruments:
2. What instrument do you play?
3. How was that to learn?
4. Which methods was used to teach you that instrument?
5. Why did you choose that instrument?
6. Do you have any hardships playing that instrument?
7. Is there an instrument you would like to play but are unable to because of your visual impairment?

If the interviewed has experience composing music
8. How did you learn to compose music?
9. Which methods was used to teach you how to compose?
10. How do you compose music?
11. Is there any tasks regarding composing that you do not feel you are able to do with the tools you have?

If the interviewed has experience with music production
11. How do you produce music?
12. How did you learn that?
13. Do you have any experience with MIDI?
If yes: 14. How do you program MIDI sound?
15. How do you design synthetic sound?
16. Is there any tasks you have a hard time with because of your visual impairment?
17. Is there any tasks you cannot perform because of your visual impairment?

If no experience working with sound
18. Do you know about one or more methods visual impaired persons work with sound and music?

Appendix B

Lifecycle Model

As it is has been chosen to use the ISO 13407 human-centered design lifecycle model it will be explained a little further in the following section.

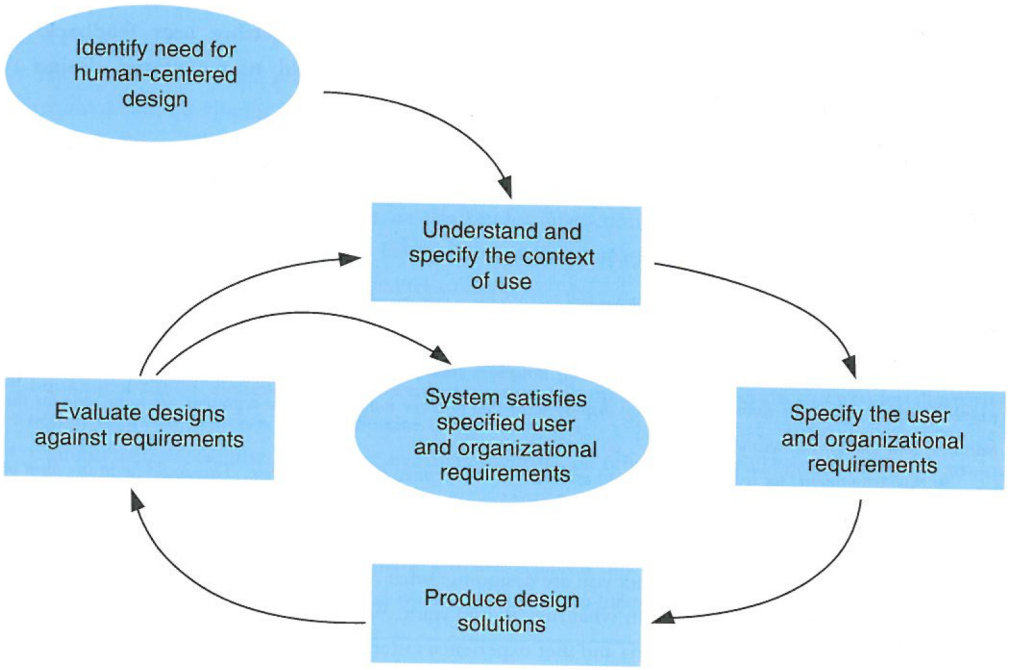


Figure B.1: The ISO 13407 model

The ISO 13407 is used when developing interactive products and gives the guidance on human-centered design activities. It shows the lifecycle when heading towards a final product. It does not specify the design approaches when leading to the final product. The model gives four principles of human-centered design[22]:

1. *"To have the user in the center of the design process and also a clear understanding of the user and task requirements. The more the user is within the design process the greater a chance for a fulfilling product is likely to happen."*

2. *"A good balance between the function of a user and the technology used."*
3. *"The importance of iterations due to the possible product solutions."*
4. *"A multi-disciplinary design whereas people with different roles is involved."*

In addition it specifies four human-centered design activities playing a central role when developing a system for users[22]:

1. *"To understand and specify the context of use."*
2. *"To specify the user and organizational requirements."*
3. *"To produce design solutions."*
4. *"To evaluate against requirements."*

One of the key points in this model is the iterations that make sure that the product meets the requirements used for making the optimal product. Lastly, the top box in the model suggest an accomulation of knowledge before starting the iteration processes.
The part of the report will list the product requirement for the design which will be based on the knowledge gained through out the sections of the analysis above.

Appendix C

Basic Theory of Ambisonics

This appendix will explain the basic theory behind Ambisonics.

The Ambisonics sound technology consist of two parts. The first part is to encode the sounds direction and amplitude and the second part is to decode this to a specific loudspeaker setup in a way that the listener perceive the sounds at a certain location. This can be either a 360 horizontal sound field, like used in this project(pantophonic systems) or a full sphere (periphonic systems) [1]. The second case can be seen in figure C.1. The encoding part is either done by recording the sound with a soundfield microphone or, like in the case of this report, assign the elevation and angle for a mono sound to the encoding equations.[15].

First order Ambisonics encoding equation consists of four variables *W,X,Y* and *Z*. Where *W* is simply the pressure of the sound and the *X,Y,Z* are vectors corresponding to the axial directions in space. This means that *W* contains all the information about the sound but nothing about it's location where for instance *X* contains the amount of sound that propagates in the X-direction. *X* will therefore contain the full signal if the sound lays directly on the x-axis(right in front or behind the listener) and no signal if the sound is to the right or left of listener and therefore directly on the y-axis. For sounds that lay in between the axes, their directional energy is split up in proportion according to their position[15]. The equation for encoding a monophonic sound into Ambisonics B-format, where *A* is the angle an *B* is the elevation is:

$$X = inputsignal * CosA * CosB$$
$$Y = inputsignal * SinA * CosB$$
$$Z = inputsignal * SinB$$
$$W = inputsignal * 0.707$$

Where *The 0.707 multiplier on W is there as a result of engineering considerations related to getting a more even distribution of signal levels within the four channels when taking live sound from a Soundfield microphone* [2]. If the sound does not lay directly on the unit sphere, distance cues have to be added[15]. For each speaker a precise proportion of each of the encoded directions is assigned, according to its position in the sound field. Any symmetrical speaker layout may be chosen from 1 to N speakers. The more speakers are used, the more precise will the illusion of the sounds location be[15].

[1] `http://www.icst.net/research/downloads/ambisonics-externals-for-maxmsp/`
[2] `http://www.york.ac.uk/inst/mustech/3d_audio/ambis2.htm`

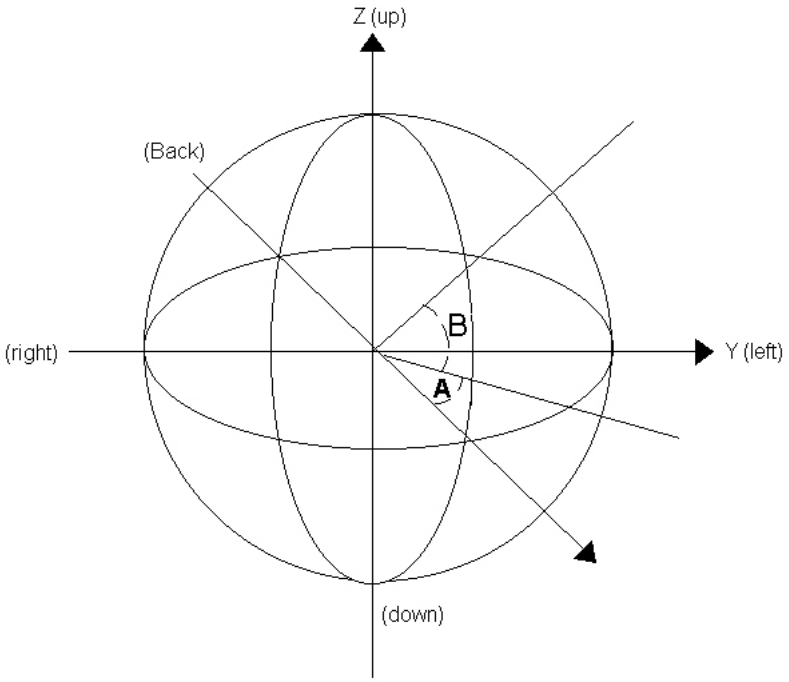


Figure C.1: Illustration of the unit sphere where A is the angle and B is the elevation

Appendix D

Usability Testing

To get a functional interface it is necessary to perform usability testing on various parts of it and apply an iterative process. Therefore a description of what a usability test is and what the purpose of one is, will be given in the following section.

Usability testing is a method where a representative user is trying a part of what could be a future product. The way of testing usability can be many, some are:

* *"testing prototypes that have only been built on paper ;"*
* *"testing prototypes that look complete but have a human behind the scenes responding ;"*
* *"testing working versions of software before it is officially released;"*
* *"testing software that has already been implemented in existing systems."*

The main goal of usability testing is to find flaws in the interface and thereby improve it by removing them. Another aspect is to find out, what is already working well.
Usability testing that takes place early in the development, tends to be exploratory and is referred as formative testing. Formative testing are used when one has a low-fidelity prototype up and running. One of the benefits of testing in such an early stage, is that users feels more comfortable by criticizing and give constructive feedback.
When one have a more developed prototype usability testing is referred as a summative test. Here the goal is to find out if the product meets the requirements of some chosen designs. Products in this stage of the process is called high-fidelity prototypes.
In the end when testing the product right before it is released, is called a validation test. This is to make sure that the product works as intended. This could for example be that 90% of the users are capable of solving a task within a specific timeframe.
A fourth type of usability testing is called comparison test and can be applied to the other three previously described. It is used to compare different design ideas, which one to choose. This will typically be used to find out which design that will benefit the product the most, and to better understand advantages and disadvantages for different design solutions.

Another main part in usability testing, and in other test methods, is the planning before the actual test. Lazar, 2006, have given a list of how to and what to when planning a usability test:

1. *"Select representative users."*
2. *"Select the setting"*

3. *"Decide what tasks users should perform"*
4. *"Decide what type of data to collect."*
5. *"Before the test session (informed consent, etc.)."*
6. *"During the test session."*
7. *"Debriefing after the session"*

All this should be planned before testing. Another discussed aspect of usability testing is, how many users a sufficient. But the folklore in HCI is, when dealing with a little project it will be sufficient with 7+/- 2 users [19], when doing the test. When dealing with such a little number of test participants, the way to make sense of the data is to use descriptive data.

D.1 Test Procedure and Result of the Usability Tests of Panning

Participant nr:

The following test is about knowing which step is playing in a step sequencer by the use of a panned sound. The sequence is 4*4 steps so 16 steps in total going from left to right, before the test starts you will hear the sequence so you know how it sounds. When the test starts you will be sitting on a chair between two speakers. When you are ready the researcher will start the sequence ad a random place in the sequence you will then follow the steps the sequence using only you ears when the sound is panned all the way to the left it is at the first step and when the sound is panned all to the right it is the last step. At some point the researcher stops the sequence and you will have to mark an X in the table below on the step you think the sequences has stopped at. This test will be done twice which is the reason why there are two tables below. At last there is one question we will like you to answer. If you have any question now you are welcome to ask.

1:

2:

Do you think there was a difference in the two test and if yes what was the difference and which of the two did you prefer?(please write below)

This test is an anonymous test and by signing this document i agree on the result of the test must be used in group 3's 4th semester project.

Please write your signature below:

_____

Test Results for the Intensity Panning Test

		The difference between the actual place and the guessed place in the panning
Participant 1	Sequential	3
	Step-based	3
Participant 2	Sequential	3
	Step-based	1
Participant 3	Sequential	0
	Step-based	0
Participant 4	Sequential	7
	Step-based	1
Participant 5	Step-based	0
	Sequential	0
Participant 6	Step-based	1
	Sequential	0
Participant 7	Step-based	0
	Sequential	1
Participant 8	Step-based	7
	Sequential	4
Average distance from the correct answer - Step-based	1,6	
Average distance from the correct answer - Sequential	2,3	

Test Results for the Ambisonics Panning Test

		The difference between the actual place and the guessed place in the panning
Participant 1	Sequential	0
	Step-based	0
Participant 2	Sequential	2
	Step-based	7
Participant 3	Sequential	2
	Step-based	0
Participant 4	Sequential	0
	Step-based	0
Participant 5	Step-based	1
	Sequential	0
Participant 6	Step-based	4
	Sequential	3
Participant 7	Step-based	3
	Sequential	3
Participant 8	Step-based	1
	Sequential	0
Average distance from the correct answer - Step-based	2	
Average distance from the correct answer - Sequential	1,3	

Appendix E

Tests

E.1 Description and Result of the Usability Test of Switches

1. Medialogy students will be used as substitutes for blind people and will therefore be blindfolded.
2. The settings will be in the group room
3. With a total of 16 switches, eight of each kind, the first participants will be testing on first sliders and then switches, the last participants will do it in reverse order. This is to avoid the possibility of learning will bias the results. On the first four of the eight switches there will be a pre-made pattern where they have to tell one which are on and which are off. Afterwards they will have to recreate that pattern on the next switches. The same will be done for the next eight switches.
4. The data collected is time. Which switches are they fastest on. After the participant is done, one will ask them which type of switches they liked the most.
5. The participant will be told on what going on, and on the same time they are blindfolded. The interface will be covered with a piece of paper so that they can not see it. When they are blindfolded the paper will be removed and the test will begin.
6. The participant are allowed to ask any questions and if they are lost one will help them guide there hands to right direction.
7. 7. Descriptive data will be used when data have been collected.

The results of test a listed below:

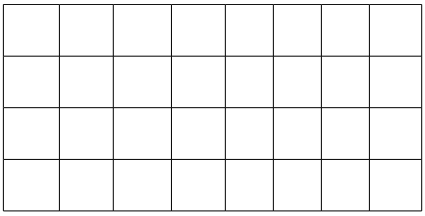
Test Results for the Test Comparing Toggle and Slide Switches

		Time before guessed the pre-made pattern	Time before recreated the pattern
Participant 1	Switches	18,6 seconds	28,2 seconds
	Buttons	29,8 seconds	53,4 seconds
Participant 2	Switches	18,3 seconds	22,5 seconds
	Buttons	27,4 seconds	7,8 seconds
Participant 3	Switches	13,6 seconds	15,3 seconds
	Buttons	17,5 seconds	18,2 seconds
Participant 4	Buttons	10,1 seconds	9,3 seconds
	Switches	11,0 seconds	9,5 seconds
Participant 5	Buttons	29,3 seconds	16,4 seconds
	Switches	28,7 seconds	13,5 seconds
Participant 6	Buttons	47,7 seconds	12,2 seconds
	Switches	55,9 seconds	12,1 seconds
Average Time Buttons		26,9 seconds	19,6 seconds
Average Time Switches		24,4 seconds	16,9 seconds

E.2 Test Procedure and Result of the Usability Test of Boundaries and Shape of Switches

Test nr :
Participant nr:

BEFORE THE TEST
The following test is to evaluate how best to create groups of buttons when you cannot actually see them. You will be blindfolded and asked to try out different layouts. The layout of the buttons will always be arranged in 4 rows and 8 columns like this:



With each type of layout you will first get some time to get accustomed with it The test continues whenever you feel you are familiar with the layout. Afterwards you will be asked to locate a button by row and column number. This will be repeated 3 times. This is not to evaluate you but the layout. When this has been done for all layouts you are asked to fill out the checkboxes and give your signature in the section “AFTER THE TEST”.

If you have any question now you are welcome to ask or else say you are ready.

AFTER THE TEST
Which one of the conditions did you prefer?

Layout A [ ]

Layout B [ ]

I preferred none of them [ ]

This test is an anonymous test and by signing this document i agree on the result of the test must be
used in group 3's 4th semester project.

Please write your signature below:

_____

Test Results for the Test of Boundaries

		Time before guessed button number 1	Time before guessed button number 2	Time before guessed button number 3
Participant 1	With boundaries	3,3 seconds	3,5 seconds	3,6 seconds
	Without boundaries	3,5 seconds	2,8 seconds	2,7 seconds
Participant 2	With boundaries	13,0 seconds	5,8 seconds	9,2 seconds
	Without boundaries	5,2 seconds	4,9 seconds	5,0 seconds
Participant 3	With boundaries	3,9 seconds	2,7 seconds	5,4 seconds
	Without boundaries	1,7 seconds	6,1 seconds	2,4 seconds
Participant 4	Without boundaries	4,4 seconds	2,5 seconds	3,1 seconds
	With boundaries	5,8 seconds	2,7 seconds	3,0 seconds
Participant 5	Without boundaries	2,5 seconds	6,3 seconds	5,5 seconds
	With boundaries	3,0 seconds	3,6 seconds	4,1 seconds
Participant 6	Without boundaries	4,3 seconds	2,8 seconds	4,2 seconds
	With boundaries	3,4 seconds	5,4 seconds	2,4 seconds
Average time for all buttons with boundaries	14,0 seconds			
Average time for all buttons without boundaries	11,7 seconds			

Test Results for the Switch Shape Test

		Time before guessed button number 1	Time before guessed button number 2	Time before guessed button number 3
Participant 1	Same shape	5,0 seconds	3,4 seconds	3,4 seconds
	Different shape	2,8 seconds	2,5 seconds	4,2 seconds
Participant 2	Same shape	4,8 seconds	5,9 seconds	3,7 seconds
	Different shape	7,0 seconds	3,6 seconds	4,2 seconds
Participant 3	Same shape	3,5 seconds	3,1 seconds	2,8 seconds
	Different shape	2,0 seconds	5,0 seconds	4,0 seconds
Participant 4	Different shape	3,9 seconds	5,1 seconds	4,7 seconds
	Same shape	4,0 seconds	3,3 seconds	2,4 seconds
Participant 5	Different shape	3,1 seconds	3,9 seconds	5,0 seconds
	Same shape	3,7 seconds	4,2 seconds	4,4 seconds
Participant 6	Different shape	2,2 seconds	5,6 seconds	4,9 seconds
	Same shape	4,2 seconds	4,5 seconds	4,6 seconds
Average time for all buttons with different shape	12,3 seconds			
Average time for all buttons with same shape	11,9 seconds			

Appendix F

Sensors Technology

F.1 Push Buttons, Toggle and Rocker Switches

Even though push button, toggle switch and rocker switch are separated into different types in the main part of this report, they will be explained as one type here, because they are used for the same purpose: interrupt the current or diverting it from one conductor to another, which means have resistance of ∞ or 0. A important characteristic of a switch is the number of *poles* and number of *throws*, for instance *double pole single throw (DPST)*(see tab F.1) The difference between the three switches discussed in the following section is their mechanical functionality and amount of poles and throws, the contact form. Although they have different types of contact forms, they all will be used as SPST or in the case of the toggle switch as SPDT. The datasheets for all components can be found on the DVD.

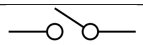
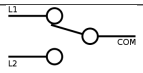
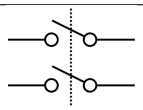
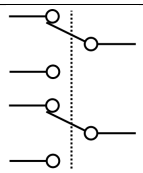
Electronics specification and abbreviation	**Description**	**Symbol**
SPST	The two terminals are either connected (ON) or disconnected (OFF).	
SPDT	A simple changeover switch: C (COM, Common) is connected to L1 or to L2.	
DPST	Equivalent to two SPST switches controlled by a single mechanism.	
DPDT	Equivalent to two SPDT switches controlled by a single mechanism: A is connected to B and D to E, or A is connected to C and D to F.	

Table F.1: Overview of the contact forms SPST, SPDT, DPST and DPDT

After having a look at the different kinds of switches, it will be discussed how the toggle switches, the rocker switches and the push-buttons are used and how they're connected.

Figure F.1: The SPPH410100 toggle switch from ALPS

For the toggle switches the *SPPH410100* from the manufacturer *ALPS*, seen in figure F.1, was chosen. Based on its contact form DPDT it is usually used to connect each of the 2 common pins to one of 2 output pins (see tab F.1).However in our case the toggle switch is used as SPDT, where *Vout* is either connected to ground via a pull-down resistor or to *Vcc*. These can easily be done by using just one line/side of the pins instead of both lines (see fig F.2). In the first case *Vout* would be 0V, because it's connected to ground and therefore LOW. In the other case, where *Vout* is connected to *Vcc* the output voltage would be 5V and therefore HIGH. Due to the amount of toggle switches and the lack of inputs on the Arduino, the toggle switches are connected to a multiplexer instead of straight to the Arduino. The Multiplexer will be explained in Appendix F.5.

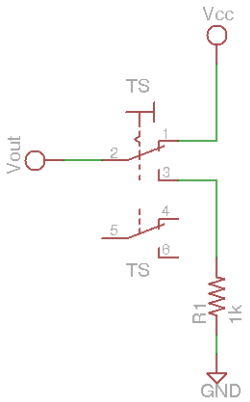


Figure F.2: Schematic of DPDT toggle switch, which is connected for SPDT usage

Figure F.3: The rocker switches FRH32A2BBBNN from CHERRY and SC791 from BAOKEZHEN

The rocker switches which were implemented are identical to the *FRH32A2BBBNN* from *CHERRY* and the SC791 from BAOKEZHEN (see fig F.3). Where the first one is a DPST and the second SPST, both are used as SPST and therefore connected in the same way. One pin is connected to *Vout* and ground via a pull-down resistor and the other pin is connected to *Vcc*. If the Rocker switch is OFF the connection between *Vcc* and *Vout* is interrupted and therefore *Vout* is just connected to ground and will have a voltage of 0V, which is represented as LOW. If the switch is ON, *Vout* and *Vcc* are connected and *Vout* will have a voltage of 5V, represented as HIGH (fig F.4). Like mentioned in the main part of the report, there is no further circuit needed and each rocker switch is connected straight to a digital input on the Arduino.

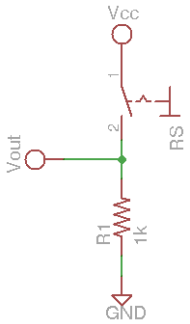


Figure F.4: Schematic of SPST rocker switch connection

Figure F.5: The R13-24A from SCI Parts

For the two implemented push-buttons the *R13-24A* from *SCI Parts* were chosen (see fig F.5). Using the SPST contact form, they connect *Vcc* with *Vout* while they are pressed and disconnect them, if they are not operated. Which means that *Vout* is only connected to ground and therefore 0V, while the push button is not operated and 5V if the button is pressed. These two states ON/OFF, 5V/0V or LOW/HIGH could be connected to the digital inputs of the Arduino. However, due to limited digital inputs it was decided to include the push buttons in a resistor ladder and therefore convert their digital states into analog voltage steps, which will be discussed in the next section of the appendix: the resistor ladder.

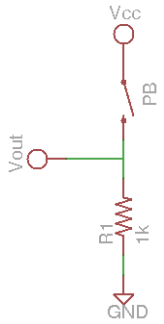


Figure F.6: Schematic of SPST push button connection

F.2 Rotary Switches and Voltage Divider

This section of the appendix will have a deeper look at the used rotary switches and how they are combined with a voltage divider.

Common rotary switches consists of a rotor (or spindle) with a small contact arm, also called spoke. An important characteristic of a rotary switches is the amount of *poles* and *ways*, where each *pole* has a fixed amount of *ways* (see fig F.7). By turning the spindle in a clockwise or anti-clockwise direction, the spoke works as a conductor between the *pole* and one of the *ways*. Therefore the *pole* is always connected to one of the *ways*.

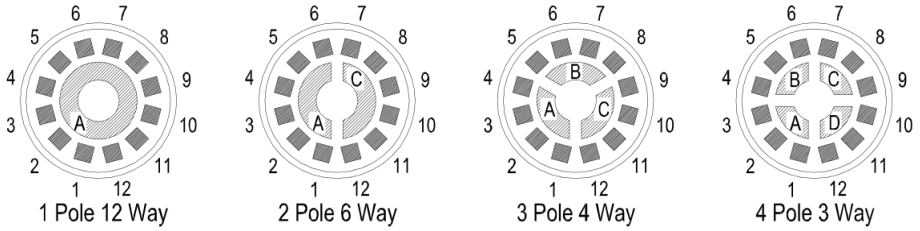


Figure F.7: Pole/Way relationship for a rotary switch with a maximum of 12 steps

Figure F.8: CK Rotary Switch from *Lorin Electronics Ltd.*

In the case of the tangible step sequencer a 3 pole 4 way CK Rotary Switch from *Lorin Electronics Ltd.* were implemented, which can be seen in fig F.8 and the datasheet can be found on the DVD. Although the chosen rotary switch has 12 available steps, it was limited to 4 steps. This can be done by changing a physical boundary inside the rotary switch. After limiting the rotary switch to 4 steps, *pole A* will be connected to *way 1,2,3 or 4* and the remaining three *ways* will stay disconnected. Connecting *Vcc* to the *pole* and each of the *ways* to a separate *Vout* as well as to ground via a pull-down resistor, one of the 4 *Vout* would be HIGH and the other three LOW. Based on this knowledge each rotary switch could be connected to 4 digital inputs of the Arduino. However, due to the lack of inputs it was decided to combine the rotary switch with a series of voltage divider and therefore use just one analog input instead.
First it will be explained what a voltage divider is and afterwards how it is combined with the rotary switch.
A voltage divider is a linear circuit, which produces a output voltage *Vout*. This output voltage *Vout* is a fraction of *Vcc*. In the case of this project a resistive voltage divider was chosen (see fig F.9), which are two resistors in series.

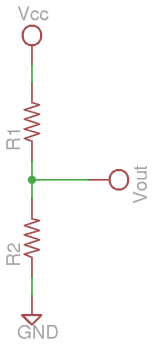


Figure F.9: Schematic of a resistive voltage divider

The amount of current which flows through each resistor in series is the same, but the voltage across every resistor varies, depending on the resistance of the resistors. The total resistance and the current can be calculated as follows:

$$R_{tot} = R1 + R2 \qquad I = \frac{Vcc}{R_{tot}}$$

The following voltage divider equation can be derived from the knowledge that the current stays the same ovder the whole circuit, and therefore $I_2 = I$, and Ohm's law which states $V_2 = I_2 * R2$. Where I_2 is the current flowing through *R2* and V_2 the voltage across *R2* and therefore $V_2 = Vout$.

$$\text{Voltage Divider Equation:} \qquad Vout = \frac{R2}{R1+R2} Vcc$$

In this formula it can be seen that *Vout* is depending on the resistance of the resistors as well as on *Vcc*, which means if we change for instance *R1* and keep the other variables the same, *Vout* will be affected in some way. This is what was done by combining the voltage divider with a rotary switch. The rotary switch switches between different resistors with different resistances, while the other values stay the same. Therefore the position of the rotary switch has direct influence on *Vout*, where each position is represented in a analog voltage. How this is done can be seen in the schematic F.10, where *GND* and *Vcc* are connected to a ground and a 5V pin of the Arduino, and *Vout* to an analog input.

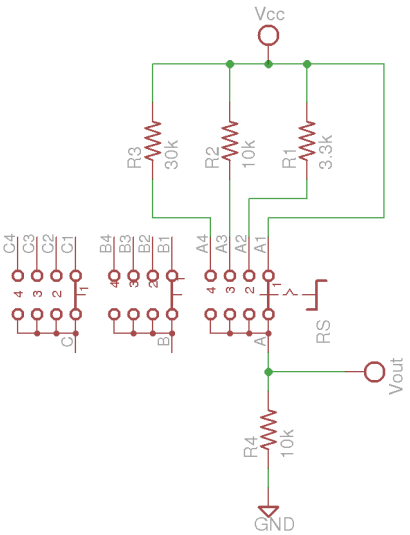


Figure F.10: Schematic of rotary switch combined with a voltage divider

To avoid ambiguous values for *Vout*, for different positions of the rotary switch, the implemented resistors *R1, R2* and *R3* have to have a resistance, that the 4 possible *Vout* have proper distance between each other. Therefore resistor sizes were chosen to change *Vout* to *Vcc*, $\frac{3}{4}Vcc$, $\frac{1}{2}Vcc$ and $\frac{1}{4}Vcc$. How these resistances were calculated can be seen below.

Position 1: $Vout = Vcc \implies$ 0 resistance and therefore no resistor needed

Position 2: $Vout = \frac{3}{4}Vcc$, $Vout = \frac{R4}{R1+R4}Vcc \implies R1 = \frac{1}{3}R4$

Position 3: $Vout = \frac{1}{2}Vcc$, $Vout = \frac{R4}{R2+R4}Vcc \implies R2 = R4$

Position 4: $Vout = \frac{1}{4}Vcc$, $Vout = \frac{R4}{R3+R4}Vcc \implies R3 = 3 * R4$

In the case of this project it was chosen to use a resistance of 10kΩ for *R4*. The Arduino supplies *Vcc* with a voltage of 5V. Based on these values, a resistance of 3.3kΩ for *R1*, 10kΩ for *R2* and 30kΩ for *R3* were implemented. This will give a voltage output of around 5V for position 1, 3.75V for position 2, 2.5V for position 3 and 1.25V for position 4. Due to signal preprocessing the Max/MSP sequencer patch will always be able to determine in which position the rotary switch is.

F.3 Resistor Ladder

Like mentioned in a previous appendix section and in the main part of the report the two push button were combined with a resistor ladder to safe two digital inputs and use just one analog input instead. The following section swill describe what a resistor ladder is and how it is used in the case of the tangible step sequencer.

A resistor ladder is combination of several resistor units, which can be used to get several digital inputs and convert them to a single output, represented as a analog voltage. However, just one of the digital inputs can be HIGH at a time. The inputs in our case are the two push buttons and therefore just one push button can be pressed at a time. Based on their task (play/pause and rewind) it can be assumed that there is no reason to press them both at the same time.

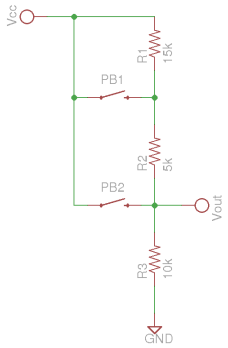


Figure F.11: Schematic of the resistor ladder including the two push buttons

Like seen in figure F.11 *Vout* is dependent from the state of the push buttons. If non of them are pressed the current flows through all resistors and based on the voltage divider, *Vout* can be calculated as follows:

$$Vout = \frac{R3}{(R1+R2)+R3} Vcc$$

If the the upper push-button *PB1* is pressed, the current flows the way with the lower resistance and therefore skip the resistor *R1* and *Vout* can be calculated as follows:

$$Vout = \frac{R3}{R2+R3} Vcc$$

For the last possibility, push button *PB2* is pressed, there is no restsitance and therefore:

$$Vout = Vcc$$

Based on the fact that the pull-down resistor *R3* has a resistance of 10kΩ, *Vcc* has a voltage of 5V and the difference between *Vout* of each possibility should be as big as possible, which means *Vout* should be $\frac{1}{3}\,Vcc$ if *PB1* is pressed and $\frac{2}{3}\,Vcc$ if *PB2* is pressed, the resistor sizes can be calculated as follows:

PB2 is pressed: $\quad Vout = \frac{R3}{R2+R3}Vcc \;\;,\;\; Vout = \frac{2}{3}Vcc \quad \Longrightarrow \quad R2 = \frac{R3}{2}$

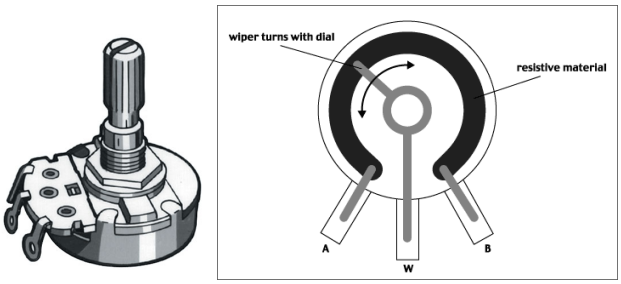
PB1 is pressed: $\quad Vout = \frac{R3}{(R1+R2)+R3}Vcc \;\;,\;\; Vout = \frac{1}{3}Vcc \;\;,\;\; R2 = \frac{R3}{2} \quad \Longrightarrow \quad R1 = \frac{3}{2}R3$

Using a resistance of 15kΩ for *R1* and 5kΩ for *R2*, *Vout* will gain 3 unambiguous voltage values, depending on which push button is pressed or if non of them are operated. *Vout* can therefore connected to analog input of the Arduino and due to signal preprocessing (see section 5.1.3) the states can be determined.

F.4 Potentiometer

The potentiometer sensor used to control the tempo or beats per minute of the product will be described in this appendix.

Since the tempo control was designed to not have any steps it was chosen to implement it with a simple rotary potentiometer. A potentiometer is a three terminal element that is made from a single piece of resistive material formed in a circular manner[26] . the middle terminal is connected to a wiper that is in electrical contact with the resistive material. This wiper is in turn connected to a axle where a knob can be mounted as depicted on figure F.12a. Since the wiper divides the single resistive element into two, a voltage divider is created which output voltage can be changed by changing the position of the wiper and therefore the relationship between the two resistances. This is shown in figure F.13.



(a) A potentiometer (b) Potentiometer schematic

Figure F.12: Picture and schematic for a potentiometer

In regards to resistive behaviour there exists two types of potentiometers. The first is the linear one where the output voltage of the potentiometer is mapped linearly to the position of the wiper. The second is mapped logarithmically to the position of the wiper. This type is often used to control volume since the logarithimc increase of volume is perceived as a linear one as described by the decibel scale. It should be noted that logarithmic potentiometers does not produce a truly logarithmic output since this would be too expensive to manufacture. Instead the output is an approximation as can be seen on the graph on figure F.13.

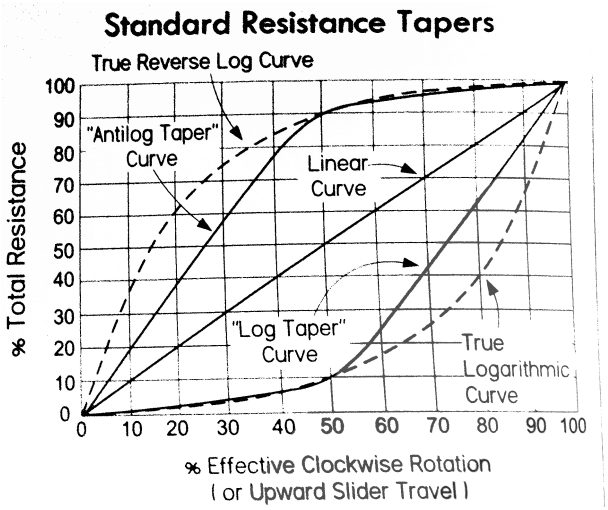


Figure F.13: Graph showing resistance for different types of potentiometers as a function of wiper position

Since it was desired to have a linear mapping between the turning of the tempo knob and the tempo a linear potentiometer was used.

According to the position of the wiper the linear potentiometer produces an output voltage in the range of 0 to *Vcc*. In the case of this project *Vcc* was 5V which meant the potentiometer could be interfaced directly to the Arduino, where the outer terminals were connected to *Vcc* and *GND* and *Vout* to a analog input (see fig. F.14).

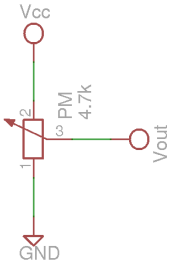


Figure F.14: Schematic of the potentiometer connection

F.5 Multiplexer

The following part of the appendix will discuss what a multiplexer is, why it is used in the case of the tangible step sequencer and how it works.
A multiplexer is a device, where one out of several analog or digital inputs can be selected and directed to one output. It can be seen as multiple input, single output switch. In the case of this project it has been used to connect all the 64 toggle buttons to the Arudino, which has a limited amount of inputs. Like discussed in the design chapter, 64 toggle switches were used to create the 4 tracks of the tangible step sequencer and therefore it was chosen to implement a 64-to-1 multiplexer.

A multiplexer, also called data selector is a digitally controlled switch, which takes input from several input lines and feeding these input lines to one output line. Depending on the amount of input lines, the multiplexer requires a fixed amount of select lines, which are used to select which input line are directed to the output line. N select lines are needed for 2^N input lines. The drum sequencer consists of 64 toggle buttons (4 tracks x 16 buttons), which requires according to the just mentioned formula six select lines.

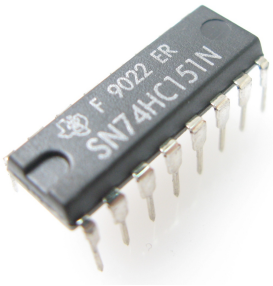


Figure F.15: Photo of a 74HC151N multiplexer

In figure F.15 a single 8-to-1 multiplexer can be seen. To implement a 64-to-1 multiplexer, nine *74HC151N* 8-to-1 multiplexer from the manufacturer *Philips* were combined. This is done by using eight single 8-to-1 multiplexer to collect all the inputs from the 64 toggle switches (8 multiplexer x 8 input lines). Each multiplexer has now eight inputs and feeds them to one output, which makes eight outputs in total. These eight outputs are connected to the eight inputs of the ninth multiplexer, which directs these eight inputs to the one final output. Figure F.16 illustrates how the just described could look like.

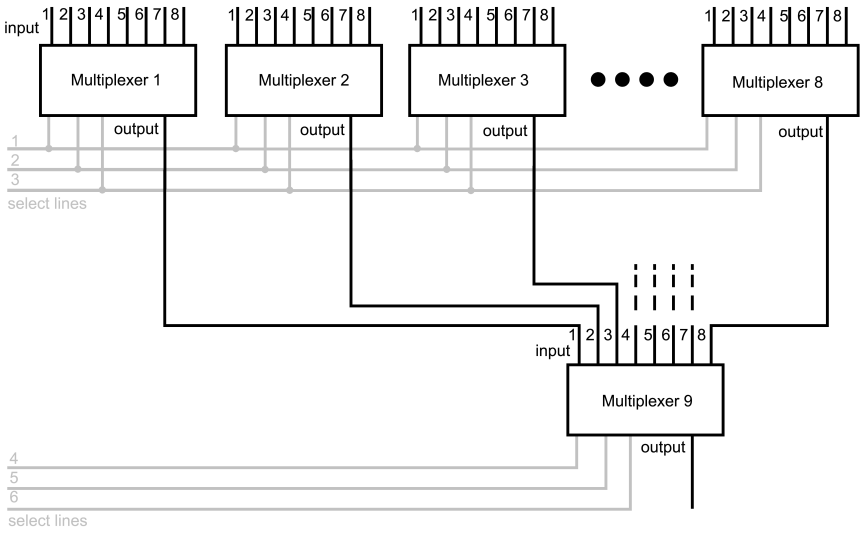
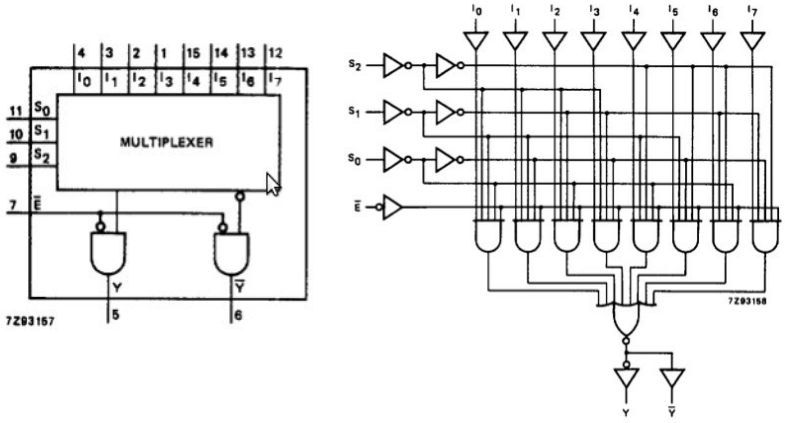


Figure F.16: Sketch of how nine 8-to-1 multiplexer are combined to one 64-to-1 multiplexer

Like mentioned before six select lines are needed to control the whole 64-to-1 multiplexers, where more precisely three of the lines control the first eight multiplexer and the other three are used to control the last multiplexer. The actual controlling is done by the Arduino. Therefore six of the digital ports of the Arduino will be used as digital outputs. The Arduino controls the multiplexer by sending digital data (HIGHs and LOWs) to the six select line. Depending on the data sent, the multiplexer directs the accordant input line to the output line (see fig F.2). For more details how the Arduino steers the multiplexers, the Ardunio Code can be found in appendix G.



(a) Functional diagram (b) Logic diagram

Figure F.17: 8-to-1 Multiplexer (Model: 74HC151)

The functionality of an multiplexer can be illustrated with the aid of an logic diagram (see fig F.17b), which shows the arrangement of the different logic gates, the multiplexer consists of. By

changing the values of the select lines S_0, S_1, S_2, the input from one input line I_0, I_1, I_2, I_3, I_4, I_5, I_6 or I_7 is allowed to pass to the output line Y and the inverted output line $\overline{Y}$. The truth table F.2 shows the eight possible combinations of S_0, S_1, S_2 and which input is passed to the output during each possibility.

Input				**Output**	
$\overline{E}$	S_0	S_1	S_2	Y	$\overline{Y}$
0	0	0	0	I_0	$\overline{I}_0$
0	0	0	1	I_1	$\overline{I}_1$
0	0	1	0	I_2	$\overline{I}_2$
0	0	1	1	I_3	$\overline{I}_3$
0	1	0	0	I_4	$\overline{I}_4$
0	1	0	1	I_5	$\overline{I}_5$
0	1	1	0	I_6	$\overline{I}_6$
0	1	1	1	I_7	$\overline{I}_7$
1	0 v 1	0 v 1	0 v 1	0	1

Table F.2: Truth table for the 8-to-1 multiplexer 74HC151

The enable input line $\overline{E}$ is used to control if the input should be connected to the output or not, which means if the multiplexer is enabled or disabled. If $\overline{E}$ is HIGH the inputs are blocked and there will be no output. On the other hand, if $\overline{E}$ is LOW, the inputs can pass to the output. For this reason, the multiplexer will be always enabled by connecting $\overline{E}$ to ground, which is done in the case of this project. This and all other connections can be seen in schematic F.18

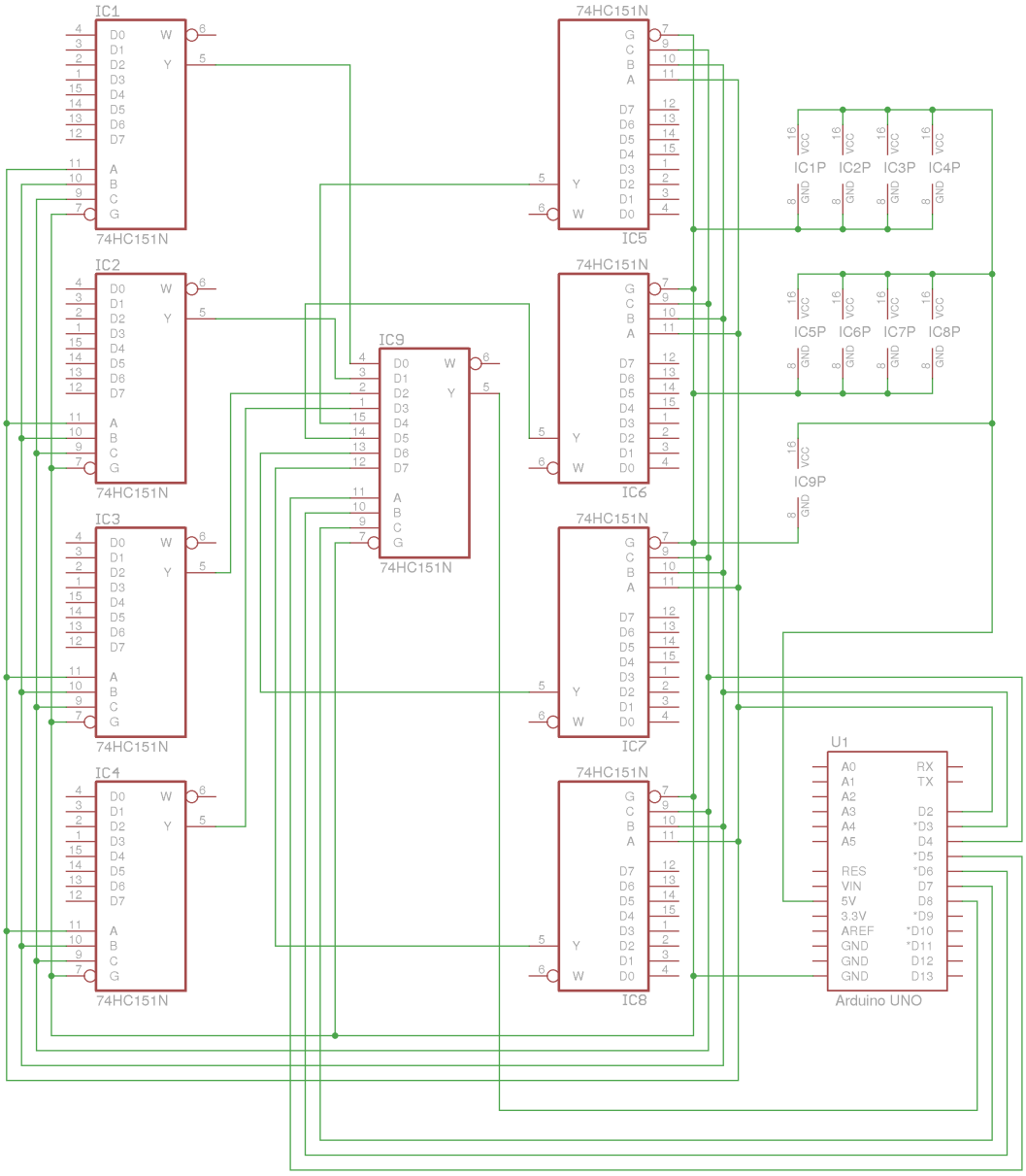


Figure F.18: Full schematic of the 64-to-1 multiplexer combining nine 8-to-1 multiplexer referred as *IC1* - *IC9* connected to the Arduino board

Appendix G

Arduino Code

This is the source code for the custom program for the Arduino that has been created in order to enable interfacing with the multiplexer described in an earlier appendix.

```
1  /*
2   *  Arduino + multiplexer 2 Max
3   *
4   *
5   *
6   *  Copyleft: use as you like
7   *  Based on a sketch by djmatic and sketch and patch by
8   *  Thomas Ouellet Fredericks  tof.danslchamp.org
9   *
10  */
11
12 int x = 0; // a place to hold pin values
13
14 //multiplexer variables
15 //the highest of the pins used to set the output from the multiplexer
16 int highestMultiOutputPin = 7;
17 //where the input from the multiplexer comes in
18 int multiplexerInputPin = 8;
19
20 int numOfButtons  = 64; //number of buttons connected to the multiplexer
21 int buttonStates[64]; //array to save the states of the buttons connected to the multiplexer
22 int numOfDigitalSensors = 5; //how many sensors connected to
23 int digitalStates[5]; //array to save the last states of sensors connected to the digital inputs
24 int lowestDigitalInputPin = 9; //the lowest pin value of the digital sensors
25 int numOfAnalogSensors  = 6; //how many sensors are connected to the analog inputs
26 int analogStates[6]; //array to save the last states of sensors connected to the analog inputs
27 int lowestAnalogInputPin = 0; //the lowest analog pin connected to a sensor
28 //if analog values change with more than this the value is sent to the MaxMSP patch
29 int analogSensivity = 5;
30
31 int currentSensor = 0; //used to loop trough all the sensors
32
33 void setup() //this function is run once when the Arduino is turned on
34 {
35
36     //setup digial pins as outputs for selection on the multiplexers
37     for(int i = highestMultiOutputPin ; i > highestMultiOutputPin - 6 ; i--)
38         pinMode(i, OUTPUT);
39
40     //setup digitial pin to take input from the multiplexer
```

```
41      pinMode(multiplexerInputPin, INPUT);
42
43      //setup other digital pins for use as inputs
44      for(int i = lowestDigitalInputPin ; i > lowestDigitalInputPin + numOfDigitalSensors ; i--)
45          pinMode(i, INPUT);
46
47      //setup analog pins as input
48      for(int i = lowestAnalogInputPin ; i > lowestAnalogInputPin + numOfAnalogSensors ; i--)
49          pinMode(i, INPUT);
50
51      Serial.begin(115200); // begin serial connection at 115200 baud (bits per second)
52  }
53
54  void loop()
55  {
56      if (Serial.available() != -1) // Check serial buffer for characters
57      {
58          char serialIn = Serial.read();//read the first character in the serial buffer
59          //if an r is recieved, the current sensor's value is sent
60          //and stored in the appropiate array
61          if (serialIn == 'r')
62          {
63        //if the sensor that should be sampled is in the range of buttons connected
64        //to the multiplexer
65              if(currentSensor < numOfButtons)
66              {
67                  //first send the signature of the button currently being sampled
68                  sendValue(currentSensor);
69
70              //then the selection pins on the multiplexer is set
71              //to the value of the variable currentSensor in binary
72              //since the value is modified it is stored in another variable
73                  int bin = currentSensor;
74                  if(bin - 32 >= 0) //should bit 5 (2    = 32) be set?
75                  {
76                      digitalWrite(highestMultiOutputPin, HIGH); //set bit 6 to high
77                      bin -= 32; //substract the value of bit 5 from the number
78                  }
79                  else
80                      digitalWrite(highestMultiOutputPin, LOW); //otherwise set bit 5 to low
81
82                  if(bin - 16 >= 0) //etc for bit 4..
83                  {
84                      digitalWrite(highestMultiOutputPin - 1, HIGH);
85                      bin -= 16;
86                  }
87                  else
88                      digitalWrite(highestMultiOutputPin - 1, LOW);
89
90                  if(bin - 8 >= 0)//etc for bit 3..
91                  {
92                      digitalWrite(highestMultiOutputPin - 2, HIGH);
93                      bin -= 8;
94                  }
95                  else
96                      digitalWrite(highestMultiOutputPin - 2, LOW);
97
98                  if(bin - 4 >= 0)//etc for bit 2..
99                  {
100                     digitalWrite(highestMultiOutputPin - 3, HIGH);
101                     bin -= 4;
102                 }
```

```
103                 else
104                     digitalWrite(highestMultiOutputPin - 3, LOW);
105
106
107                 if(bin - 2 >= 0)//etc for bit 1
108                 {
109                     digitalWrite(highestMultiOutputPin - 4, HIGH);
110                     bin -= 2;
111                 }
112
113                 else
114                     digitalWrite(highestMultiOutputPin - 4, LOW);
115
116                 if(bin - 1 >= 0)//etc for bit 0
117                 {
118                     digitalWrite(highestMultiOutputPin - 5, HIGH);
119                     bin -= 1;
120                 }
121                 else
122                     digitalWrite(highestMultiOutputPin - 5, LOW);
123
124                 //read the value from the multiplexer and send it
125                 sendValue(digitalRead(multiplexerInputPin));
126                 //save the state of the button
127                 buttonStates[currentSensor] = digitalRead(multiplexerInputPin);
128
129             }
130             //if the current sensor to sample is in the range of digital sensors
131             else if(currentSensor < numOfButtons + numOfDigitalSensors)
132             {
133                 //get the current digital sensor
134                 int currentDigitalInputPin = currentSensor - numOfButtons;
135                 //send the signature for this sensor
136                 //all input in the 70ies range are treated as digital inputs in MaxMSP
137                 sendValue(70 + currentDigitalInputPin);
138                 //save the state of the sensor
139                 digitalStates[currentDigitalInputPin] =
140                 digitalRead(lowestDigitalInputPin + currentDigitalInputPin);
141                 //at the end is the state of the sensor sent to MaxMSP
142                 sendValue(digitalRead(lowestDigitalInputPin + currentDigitalInputPin));
143             }
144             //at last, if the current sensor is in the range of analog sensors
145             else if(currentSensor < numOfButtons + numOfDigitalSensors + numOfAnalogSensors)
146             {
147                 //get the current analog sensor
148                 int currentAnalogInputPin = currentSensor - numOfButtons - numOfDigitalSensors;
149                 //send the signature for this sensor
150                 //all input in the 80ies range are treated as analog inputs in MaxMSP
151                 sendValue(80 + currentAnalogInputPin);
152                 //save the value of the analog sensor
153                 analogStates[currentAnalogInputPin] =
154                 analogRead(lowestAnalogInputPin + currentAnalogInputPin);
155
156                 //send the value of the sensor
157                 sendValue(analogRead(lowestAnalogInputPin + currentAnalogInputPin));
158             }
159
160         Serial.println(); // Send a carriage returnt to mark end of serial transmission
161
162         //notify serial comm (or XMLSocket for Flash) that the comunication is over
163         // doing that by writing byte 0 on end
164         // must mark with byte 0 at end to have normal serial readers to recognize...
```

```
165             Serial.print(0, BYTE);  // for ARDUINO IDE > 10
166
167             //if all sensors have been read
168             if(currentSensor >= numOfButtons + numOfDigitalSensors + numOfAnalogSensors -1)
169                //reset currentSensor so the first sensor is read next time an r is sent
170                currentSensor = 0;
171             else
172                //otherwise increase currentSensor by one
173                //so the next sensor is read the next time an r is recieved
174                currentSensor++;
175         }
176
177         //if an u is received it is to update the state of the sensors and transmit new values
178         if (serialIn == 'u')
179           {
180               //used to see if we should brake out of the while loop beneath
181               boolean breakFree = false;
182               while(true) //for as long as possible
183               {
184                   //is the current sensor one of the buttons connected to the multiplexer?
185                   if(currentSensor < numOfButtons)
186                   {
187                       //since the value is modified it is stored in another variable
188                       int bin = currentSensor;
189                       if(bin - 32 >= 0) //should bit 5 be set?
190                       {
191                           digitalWrite(highestMultiOutputPin, HIGH);
192                           bin -= 32;
193                       }
194                       else
195                           digitalWrite(highestMultiOutputPin, LOW);
196
197                       if(bin - 16 >= 0)//should bit 4 be set?
198                       {
199                           digitalWrite(highestMultiOutputPin - 1, HIGH);
200                           bin -= 16;
201
202                       }
203                       else
204                           digitalWrite(highestMultiOutputPin - 1, LOW);
205
206                       if(bin - 8 >= 0)//should bit 3 be set?
207                       {
208                           digitalWrite(highestMultiOutputPin - 2, HIGH);
209                           bin -= 8;
210                       }
211                       else
212                           digitalWrite(highestMultiOutputPin - 2, LOW);
213
214                       if(bin - 4 >= 0)//should bit 2 be set?
215                       {
216                           digitalWrite(highestMultiOutputPin - 3, HIGH);
217                           bin -= 4;
218                       }
219                       else
220                           digitalWrite(highestMultiOutputPin - 3, LOW);
221
222
223                       if(bin - 2 >= 0)//should bit 1 be set?
224                       {
225                           digitalWrite(highestMultiOutputPin - 4, HIGH);
226                           bin -= 2;
```

```
227                     }
228
229                     else
230                         digitalWrite(highestMultiOutputPin - 4, LOW);
231
232                     if(bin - 1 >= 0)//should bit 0 be set?
233                     {
234                         digitalWrite(highestMultiOutputPin - 5, HIGH);
235                         bin -= 1;
236                     }
237                     else
238                         digitalWrite(highestMultiOutputPin - 5, LOW);
239
240                     if(digitalRead(multiplexerInputPin) != buttonStates[currentSensor])
241                     {
242                         breakFree = true; //save that the while loop should be exited
243                         buttonStates[currentSensor] = digitalRead(multiplexerInputPin);
244                         sendValue(currentSensor); //first send signature of the current button
245
246                         sendValue(digitalRead(multiplexerInputPin));
247                         Serial.println();//Send a carriage returnt to mark end of pin data.
248                         Serial.print(0, BYTE);//for ARDUINO IDE > 10
249                     }
250                 }
251                 else if(currentSensor < numOfButtons + numOfDigitalSensors)
252                 {
253                     //get the current digital sensor
254                     int currentDigitalInputPin = currentSensor - numOfButtons;
255                     //send the signature for this sensor
256                     //all input in the 70ies range are treated as digital inputs in MaxMSP
257                     //read the state of the sensor
258                     int readData = digitalRead(lowestDigitalInputPin + currentDigitalInputPin);
259                     //If the state has changed:
260                     if(readData != digitalStates[currentDigitalInputPin])
261                     {
262                         //Store the state of the sensor
263                         digitalStates[currentDigitalInputPin] = readData;
264                         //send over the signature of the sensor
265                         sendValue(70 + currentDigitalInputPin);
266                         sendValue(readData);//send the state of the sensor
267                         Serial.println();//Send a carriage return to mark end of pin data.
268                         Serial.print(0, BYTE);// for ARDUINO IDE > 10
269                         breakFree = true; //break free of the loop
270                     }
271             }
272             //if the current sensor is one of the analog values
273             else if(currentSensor < numOfButtons + numOfDigitalSensors + numOfAnalogSensors)
274             {
275                 //get the current analog sensor
276                 int currentAnalogInputPin = currentSensor - numOfButtons - numOfDigitalSensors;
277                 //read its value
278                 int readData = analogRead(lowestAnalogInputPin + currentAnalogInputPin);
279
280                 //if its value has changed enough from the last value
281                 //to not just be caused by noise
282                 if(readData - analogSensivity > analogStates[currentAnalogInputPin] ||
283                    eadData + analogSensivity < analogStates[currentAnalogInputPin])
284                 {
285                 //save the new value
286                     analogStates[currentAnalogInputPin] = readData;
287                     //send the signature
288                     sendValue(80 + currentAnalogInputPin);
```

```
289                         sendValue(readData); //send its value
290                         Serial.println();// Send a carriage returnt to mark end of pin data.
291                         Serial.print(0, BYTE);  // for ARDUINO IDE > 10
292                         breakFree = true; //break free of loop
293                     }
294                 }
295
296                  //if all sensors has been run trough
297                  if(currentSensor >= numOfButtons + numOfDigitalSensors + numOfAnalogSensors -1)
298                  {
299                     //reset back to sample the first sensor when the next 'u' is recieved
300                     currentSensor = 0;
301                     breakFree = true;
302                  }
303                  else //else go to the next sensor and continue the loop
304                     currentSensor++;
305
306                 if(breakFree) //if a sensor value has been sent we break free
307                   break;
308             }
309         }
310     }
311 }
312
313 //function to send an integer value followed by a space
314 void sendValue (int x)
315 {
316     Serial.print(x); //print value
317     Serial.print(32, BYTE); //print a space, the BYTE parameter means the 32 is sent as a char
318 }
```

Appendix H

Final Interview Guide

Test guide for the 18th of may.
The test will be a summative test whereas there will be recorded qualitative data because of the number of participants.
We want to test on utility and satisfaction.
They will get as much as they need to familiarize themselves with the interface and will thereafter be interviewed:

"what is your general thought about the interface?"

Satisfaction:

Buttons/Switches:
"If we starts with the button that you have used to activate the sounds on each track, was they easy navigate amongst?"

If yes: *"What made them easy to navigate amongst? Were they fast to operate? Could it be easier?"*
If no: *"What made it difficult? How can we improve it?"*

Did you use the rotary switches used for changing the sound on each track?"

If yes: *"Were they nice to operate? Could they be made better?"*
If no: *"How come?"*

What about the the pause/play and rewind buttons, did you use them?"

If yes: *"Were they easy to operate? Could they be made better?"*
If no: *"How come?"*

The buttons used for muting each track, did you get to use them?"

If yes: *"Were they easy to operate? Could they be made better?"*
If no: *"How come?"*

The button used for switching panning on and of, did you get use it?"

If yes: *"Was it easy to handle? Could it be made better?"*
If no: *"How come?"*

"The potentiometer used for changing the speed, did you use that?"

If yes: *"Was it easy to handle? Could it be made better?"*
If no: *"How come?"*

Boundaries:
"Did you get to feel the boundaries used on the interface?"

If yes: *"Could they be used? Could they be made better?"*
If no: *"Try to feel them, do you think that they will be useable or are they redundant? Is there other possibilities of guiding blind people around on an interface?"*

Textures:
"Did you feel the different textures used on the tracks?"

If yes: *"Were they good? Could they be made better?"*
If no: *"Should they be removed? Is there another possibility that could make it easier to indicate where on the interface you are?"*

Panning:
"Did you think that the panning worked as intended?" (a little test could be executed here to see how precise it is)

If good: *"What made it good? Could it be made better?"*
If bad: *"Should it be removed? Is there a better way to indicate where on the track you are?"*

Useful?
"Is it a product that you will be able to use for something?"

In the end:

"Is there any suggestions that you think would make the product better, it can be everything?"