LEIBNIZ UNIVERSITÄT HANNOVER
Faculty of Electrical Engineering and Computer Science
Human-Computer Interaction Group

CASUAL INTERACTION WITH A BRACELET

A Thesis presented for the degree of Master of Science

by
KAROLINE BUSSE
December 2014

First Examiner : Michael Rohs, Prof. Dr.
Second Examiner : Franz-Erich Wolter, Prof. Dr.
Supervisor : Henning Pohl, M.Sc.

ABSTRACT

Short summary of the contents in English. . .

ZUSAMMENFASSUNG

Kurze Zusammenfassung des Inhaltes in deutscher Sprache. . .

CONTENTS

1 INTRODUCTION 1
2 SCENARIO 3
3 RELATED WORK 5
4 THE BRACELET 7
4.1 Manufacturing Techniques 7
4.1.1 Computer Aided Design 7
4.1.2 3D Printing 8
4.1.3 Silicone Casting 9
4.2 Design Process and Prototype Manufacturing 10
4.2.1 Rigid Designs 10
4.2.2 Segmented Designs 12
4.2.3 Silicone Bracelet 14
4.2.4 One-Piece Silicone Bracelet 15
4.3 Teensy Development Board 16
4.4 Touch Slider 16
4.5 3D Accelerometer 17
4.6 Visual Feedback 18
5 INTERACTIVE LIGHT SOURCE 21
6 INTERACTION 23
6.1 Pairing the Bracelet with a Light Source 23
6.2 Switching the Light Source on and off 23
6.3 Adjusting the Light Source's Brightness 23
6.4 Changing the Lighting Mood 24
6.5 Precise Touch Input for Color Change 24
6.6 Template picking by Gesture Recognition 24
6.7 Presets and Configuration 29
7 EVALUATION 31
7.1 Study Design 31
7.2 Results 31
7.3 Discussion 31
8 CONCLUSION AND FUTURE WORK 33

BIBLIOGRAPHY 35

LIST OF FIGURES

Figure 1 Shore hardness scales and everyday examples 10
Figure 2 First rigid design with uniform thickness 11
Figure 3 Second design featuring a tapered shape 12
Figure 4 Third design with removable lid 12
Figure 5 Multi-part design 13
Figure 6 Block diagram of the MMA8652FC digital accelerometer. [10] 17
Figure 7 Data transfer on the Inter-Integrated Circuit (I^2C) bus. [25] 18
Figure 8 The interactive light source, encased. 21

LIST OF TABLES

Table 1 Test setups for Shore hardness types 9
Table 2 Tap detection configuration 29

LISTINGS

Listing 1 Resampling of a points path into N evenly spaced points 24
Listing 2 Rotation of points so that their indicative angle is at 0° 25
Listing 3 Scaling to reference cube and translation to origin 25
Listing 4 Matching candidate gesture against every template 25

Listing 5 Three-dimensional Golden Section Search for finding the best angle between a candidate gesture and a template 27

ACRONYMS

STL Sterolithography

HCI Human-Computer Interaction

IR Infrared

LED Light Emitting Diode

SPP Serial Port Profile

CAD Computer-Aided Design

USB Universal Serial Bus

PWM Pulse-width Modulation

MSE Mean Square Error

GSS Golden Section Search

I^{2}C Inter-Integrated Circuit

1 INTRODUCTION

2 SCENARIO

> Alice comes home from grocery shopping. While carrying her shopping to the kitchen, she quickly turns on the light by simply touching her bracelet.
>
> After storing everything, she decides to relax in the living room by reading a book. After turning the kitchen lights by covering the bracelets with her hand, she makes herself comfortable on the sofa and picks up the book. To feel more comfortable, Alice dims the living room light a little by touching the bracelet rim and tilting her wrist, just as she was turning a dimmer knob. The living room lights dim accordingly, so she doesn't need to look any closer at the bracelet.
>
> Later that day, Alice prepares dinner and the dining table in the living room. She uses the bracelet's touch controls to specify the exact setup for each light in the room. Alice saves three lighting setups together with unique activation gestures: A warm setting for their guests' arrival and enjoying the welcoming drinks, a well-matched dining setup with focus on the table and foods, and a darker, colorful mood for drinking cocktails after dinner. Changing between these presets with hand gestures allows Alice to focus on her guests and on the meal instead of wasting time and focus by fiddling with wall panels or switches.

This scenario illustrates three different levels of casual interaction with a device: A simple on/off function by covering the bracelet's touch surface, intuitive and eyes-free brightness dimming by touching the bracelet and tilting the wrist, color mood change by precise touch and a fine-tuned hue setup using focused touch interaction. In addition, three hand gestures are available to easily access frequently used setups like bright white work light or slightly dimmed relaxing illumination.

In order to enable the interactions mentoined above, the device requires certain features. First, it should stay where it is needed without encumbering the user. Typical remotes or smartphones occupy at least one hand for every interaction they offer. This is not desired, so traditional hand-held devices do not fit the scenario presented above. Instead, there is the need for a wearable device that is attached to the body without getting in the way during everyday activities.

Second, touch-free interaction (i.e. control by gestures) should be possible with respect to the casual interaction scenarios pictured above.

Capacitive touch input offers flexibility and versatility compared to traditional keypads and enables gestured touch input. In addition, the required hardware should be kept on a low cost level and encompass only needed components to keep energy consumption at a minimum, since every recharge procedure is cumbersome to the user.

A bracelet-typed device can fulfill the requirements stated above. It is slim and doesn't encumber the user, so it can be worn on the arm all day.

3 RELATED WORK

Casual interaction, which is the core topic of this thesis, has been explored by other researchers.

Pohl and Murray-Smith have researched casual interaction with everyday electronics like smartphones. They coined the term "casual interaction" in contrast to focused interaction, and described that in many situations in daily life, the user would not receive periodic feedback from the interacting device or not interact with it in a constant, uninterrupted fashion due to physical, social or mental reasons like wearing gloves in the winter or being exhausted after a day of work. Instead of giving up control over the device in such situations, the user should be able to control the level of focus to which they would like to engage [27]. The underlying distinction between foreground and background interaction in context of technology has been introduced by Buxton in 1995, where he described improvements on telemetry systems [5].

Interaction with intelligent lighting in public space as well as in the arts has been researched by Seitinger et. al. [32] [31].

Wearable electronic devices have been very present in the recent years. From activity trackers like the Fitbit bracelets [8] to smart watches that are either universal like the Pebble [26] or bound to a certain phone like Samsung's Galaxy Gear [30], various devices have aimed for a spot on the user's wrist. More recently, attempts to add "social" features to wrist-worn smart devices have been made. The iBand concept by Kanis et. al attempts to augment virtual social networks by merging their information about a person into real-world interactions with people [13]. As a commercial product, the Razer Nabu smart band can exchange contact information on a handshake among other features [28].

Controlling devices in the smart home is mostly realized by trigger-action programming as discussed by Ur et. al. [37], and enabled using the IFTTT environment[12] or customized applications for smartphones or the *Android Wear* middleware [11].

Regarding new shapes and configurations of wrist-worn wearables, research has come up with concepts involving multiple displays that form the *Facet* bracelet [21], gesture-only input for interaction with the *Gesture Watch* [15], the *Snaplet* flexible touch display that changes its application according to its current shape [36] or a tiny worn sensor that uses the surrounding skin on the arm or any other body part as an input canvas for touch and gesture input [24].

For gesture recognition, two major algorithms that originate from the field of machine learning are very present in research projects. The Gesture and Activity Recognition Toolkit *GART* has been developed by Lyons et. al. and is based on Hidden Markov Models [21], it has been used by the aforementioned *Gesture Watch*[15] and the *AirTouch*, an around-device gesture recognition device that gives tactical feedback [19]. Several other gesture recognizing wearables implement Dynamic Time Warping, another algorithm from the field of machine learning [4] [20]. In contrast to those computation-heavy and complex algorithms, Wobbrock et. al. have developed a simpler, geometry-based gesture recognizer for touch screen interaction [38] that was extended to 3D accelerometer data by Kratz and Rohs [18]. The recognizer was designed as easy to implement and functional even on embedded devices with limited computation capabilities.

4 THE BRACELET

The interactive bracelet consists of several touch sensors, a motion sensor, a status Light Emitting Diode (LED) and a Bluetooth communication module, all powered by an ARM microcontroller. The bracelet's design focuses on wearing comfort, low weight and small error of unintended activation.

4.1 MANUFACTURING TECHNIQUES

All bracelets produced for this thesis were designed using a 3D Computer-Aided Design (CAD) modeler. The first prototypes were 3D printed rigid bodies to wear as armcuffs, but those designs turned out to be too inflexible and uncomfortable to wear. A different material for the bracelets was needed, so a transition from printing to casting with liquid silicone took place. The molds are printed, assembled, filled with silicone and usually destroyed while retrieving the finished cast.

In this section, the various methods and tools used for designing and manufacturing the bracelet prototypes will be explained in detail.

4.1.1 *Computer Aided Design*

In the beginning of a design iteration, a model of the desired object is drafted with a 3D CAD program. For all designs in the context of this thesis, the open source CAD modeler FreeCAD[9] was used. This program classifies itself as a general purpose 3D parametric modeler. A typical work-flow when designing a bracelet is as follows:

A bracelet's model is based on a circumference sketch. As the human wrist does not follow a circular shape, all designs made for this thesis are based on an oval circumference. In FreeCAD, this is realized by a composition of arcs and straight lines. It is important to constraint the sketch with radii, lengths and angles as well as symmetries or perpendicular constraints until no degrees of freedom are left and thus the model is fully constrained. Only fully constrained models can be successfully exported for printing.

In the next step one or more profile cuts are added to various points along the circumference. They define the thickness and shape of the bracelet at said points. For increased comfort, the bracelet should be as slim as possible, especially on the "backside" below the palm. In more complicated designs, an alternative to complex sketches for the profiles is the creation of multiple, overlaid sketches as a preparation for a subtraction operation in the later process. All profiles need to be fully constrained as well.

After the circumference as well as one or more profiles are added to the design, a first solid is created by sweeping the profile(s) around the circumference curve. This operation creates a basic shape that can be refined further on, typically by chamfering or filleting the edges with respect to increased wearing comfort. If a complex shape is desired, multiple sweeps can be generated and used in boolean operations such as union or subtraction. This is the usual approach for most bracelet designs.

The last step in the design process is usually the finishing of the CAD model. In the bracelet case, this translates to edge smoothing with chamfer or fillet tools. Smoothed edges increase the overall wearing comfort of a bracelet, so they are very desired on edges that contact with the skin.

The finished designs are then exported as mesh files (usually in Sterolithography (STL) format) for printing.

4.1.2 *3D Printing*

All prints were manufactured using a ProJet360 3D printer[2]. When printing an object, the print bed is filled first with a base layer of powder. The powder used by the printer is called VisiJet PXL, a plaster-like substance. After filling the bed, a standard inkjet printhead is cleaned and prepared for the printing process. Immediately after this setup is complete, the additive manufacturing process starts.

The model is created by printing the binder fluid onto the plaster. After each printed layer, a new thin layer of powder is added to the print bed. The ProJet360 allows for a layer thickness of 0.1 mm[1]. This allows even delicate structures without any additional supports, since the printed object is surrounded and therefore supported by plaster powder during the production process. The only drawback of this printing process is that it doesn't support closed, hollow objects since there is no possibility of removing the enclosed excess powder after the process has finished. When designing models for 3D printing, this constraint has to be kept in mind.

The finished object is then carefully removed from the build bed and any excess powder is gently brushed or blown off. The printer offers a cleaning chamber with a pressurized air pistol and a vacuuming system to assist in that task. Without further hardening, the objects are very fragile and easy to break, even with the pressurized air pistol included in the printer. In order to drastically increase the strength of the prints, they are infiltrated with a fluid after they were thoroughly cleaned. Prints produced by the ProJet 360 can be infiltrated with one of three different substances with varying characteristics: The ColorBond "instant-cure infiltrant", the two-part Strength-

TYPE	CONFIGURATION	DIAMETER	SPRING FORCE
A	35°	1.4 mm	8.06 N
D	30°	1.4 mm	44.46 N
OO	1.2 mm in spherical radius	2.4 mm	1.11 N

Table 1: Test setups for Shore hardness types A, D, and OO[**?**]

Max infiltrant "ideal for functional models", and the Salt Water Cure "eco-friendly and hazard-free infiltrant"[1]. All prints produced for this thesis were infiltrated with ColorBond.

The infiltration step adds strength and hardens the material, resulting in a sturdy printed object. However, the objects created with this technique are very rigid and any bending load might break them easily. Wall strengths of 1.5 mm and up have been proven sturdy enough for a bracelet shape, although this also depends on the object geometry.

4.1.3 *Silicone Casting*

Another manufacturing process for bracelet prototypes used in this thesis is liquid silicone casting. Two different types of silicone were used for making various bracelet prototypes, both from manufacturer Smooth-On: Sorta-Clear 37 and Mold Star 15 Slow.

The most important characteristic for silicone in prototype production is the hardness, measured in Shore (after Albert F. Shore) or Durometer. It measures the indentation of a material with a special device which is also called Shore Durometer. It consists of a hardened steel rod with a finer tip and is available in two versions, since there are two different scales for Shore hardness (cf. table 1). The Shore A scale is designed for softer materials and the Shore D scale for harder ones, but they do overlap, so a material classified in Shore D hardness is not necessarily harder than another material classified in Shore A hardness. Each scale ranges from values 0 to 100, higher numbers indicate higher material resistance. The Shore hardness is specified in EN ISO 868.

When dealing with silicone, the Shore A scale is sometimes too "hard" for soft rubbers. Another standard (ISO 7619????) therefore specifies twelve different Durometer types, where the OO scale is commonly used for soft silicones. Figure 1 shows the three Shore hardness scales A, D, and OO, as well as some examples for everyday objects and their corresponding Shore values.

The silicone rubbers used for casting bracelet prototypes are both located on the Shore A scale. The softer Mold Star 15 Slow has a Shore A hardness of 15 that could be roughly compared to that of a rubber

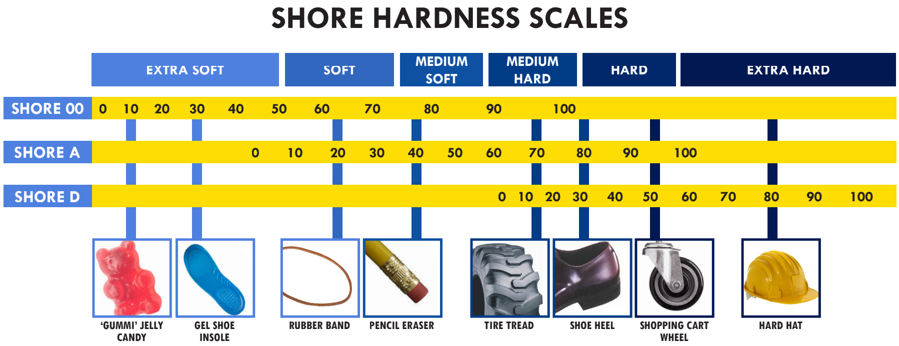


Figure 1: Shore types A, D, and OO in comparison with everyday examples of various hardnesses.[**?**]

band according to figure 1, while the slightly harder Sorta-Clear 37 has a Shore A rating of 37, similar to that of a pencil eraser[33] [34].

Both silicone products consist of two compounds that have to be added up and stirred before casting. While the Mold Star silicone has a rather low viscosity, casting the Sorta-Clear requires careful mold design, since it does not distribute well and is rather viscous. A vibrating table can help in filling the mold completely, but nonetheless were casts with the Sorta-Clear silicone much less fruitful, especially for the detailed molds of the one-piece designs (cf. section **??**).

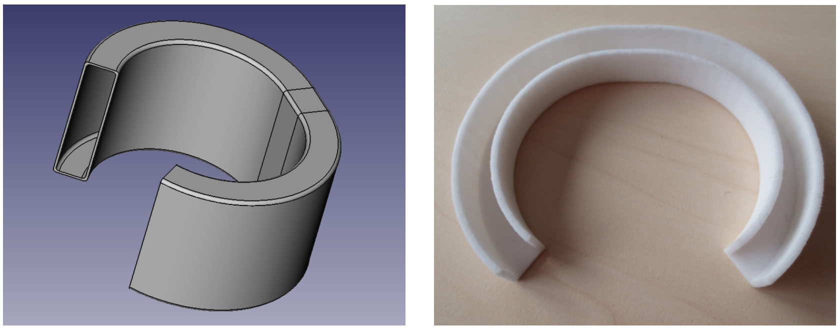
4.2 DESIGN PROCESS AND PROTOTYPE MANUFACTURING

The design process for the interactive bracelet presented in this thesis went through different stages. At first, a 3D printed casing for the electronics was favored, but turned out to be too inflexible, too fragile and even hindering while worn on the wrist. Later on, a cast silicone bracelet turned out to be more comfortable for the user. The different prototypes are explained in detail in the following sections.

4.2.1 *Rigid Designs*

The first approach that comes to mind when thinking about bracelet design is a cuff-like, rigid shape. From theCAD point of view, the first bracelet prototype consists of a single rectangular profile rotated around a oval curve which was derived from a measured wrist. The bracelet's inside is hollow, in order to store all the electronic components. The cuff's gap was just large enough for the wrist to fit through, although in reality, this lead to light scratches on the skin in combination with the rough texture of the printed material. In addition, the uniform thickness made this first prototype uncomfortable to wear, especially at the open ends.

To summarize, the uniform thickness made the bracelet feel uncomfortable on the wrist and the material felt unfriendly to the skin. So this fist prototype had some clear downsides that were eliminated in the next iteration.



(a) CAD view (b) Printed bracelet

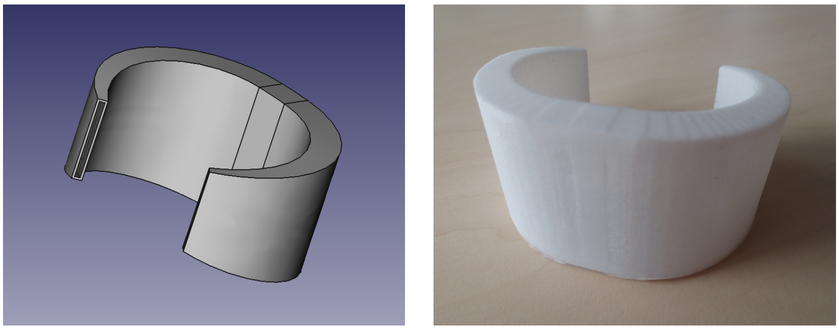
Figure 2: First rigid design with uniform thickness

In the following iteration, a bracelet of varying thickness was designed to make wearing the prototype more comfortable. The part on top of the wrist was designed to have the highest thickness, since this spot is only rarely disturbing in typical wrist movements and users are likely accustomed to some extra mass on this spot from wearing wristwatches. As the bracelet leads along the arm, its thickness decreases and shrinks to a minimum at the cuff gap. Decreasing the thickness from XX cm to YY cm made the look more appealing and wearing a little less encumbering. At the same time, the wall strength was decreased from X.Y mm to 0.7 mm, which made it very fragile in fabrication and usage, both prints broke during post-processing. Additionally, the possibility of storing electronic components inside the bracelet decreased, since smaller ends featured smaller entrances to the hollow inside. Wearing the cuff while working on a PC felt only slightly uncomfortable, but twisting the hand was still encumbered by the tight-fitting bracelet.

All in all, a tapered shape felt more comfortable, but the downsides of the material were still present and the improved shape led to less practicability from the engineering point of view. A design goal for future prototypes derived from this prototype was adding more space between the arm and the bracelet to ensure better comfort while wearing it.

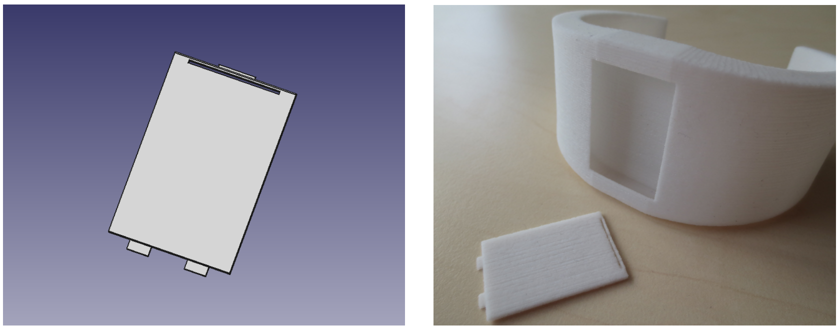
A modified design of the aforementioned prototype featured a removable lid since the tapered shape made it hard to access the inside space of the bracelet. The lid design was inspired by battery case covers commonly found in remote controls or small electronic devices. It features two rabbets on the one side that support the lid in its place and another, smaller rabbet with a cavity in the material right next to it, so the rabbet can snap just inside the lid gap when it closes.

It turned out that the lid features were designed too fine, especially the flexible part was too thin to work as intended. The lid had to



(a) CAD view (b) Printed bracelet

Figure 3: Second design featuring a tapered shape



(a) Lid in CAD view (b) Printed bracelet

Figure 4: Third design with removable lid

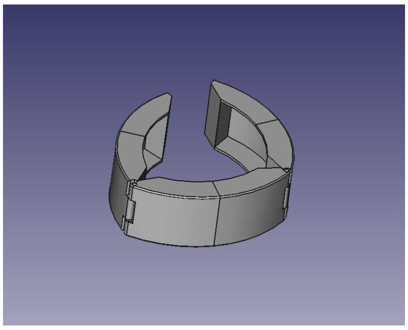
be opened and closed very cautiously and overall, the construction seemed not reliable for daily use. In addition, the rigid shape still lead to clumsiness in wearing the bracelet, so the whole concept of rigid bracelets was left behind.

4.2.2 *Segmented Designs*

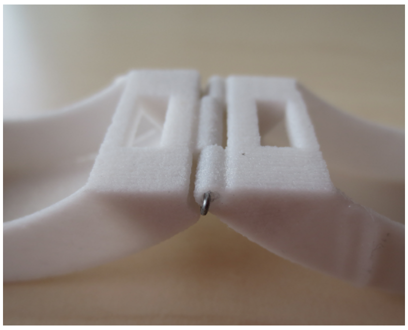
The search for a more flexible printed bracelet shape lead to an entry for an activity bracelet design contest by Daniel Muschke on a 3D printing template exchange site called *GrabCAD* [23]. Muschke created a CAD file for a bracelet consisting of three segments that are connected by slow hinges, i.e. hinges with great resistance that do not move if no external force is applied. This design considered the electronic components like a micro-controller and a micro Universal Serial Bus (USB) port which made a good start for further modification, but unfortunately the file format made it impossible to alter the design in detail. A print was possible since STL files were included in the upload. However, the design turned out to be too small to be actually wearable, but it demonstrated that printed hinges work well with a pivot made from wire. The style felt more comfortable to wear than the previous prototypes and felt leaner on the wrist than the rigid prototypes. Overall, the printed design looked promising so the idea of a multi-part bracelet was investigated further.

Recreating Muschke's design was not as easy as planned, since the parametric modeling approach on this segmented shape was very different in comparison with previous designs. The first prototypes were based on a wrist-like oval circumference curve, while the multi-part design originated from a partitioned circle, since all parts should have the same dimensions in length and curvature. Adjusting the part size to the designated wrist turned out to be difficult, and prints of the design were either too small or too big when printed.

The bracelet segments are hollow and open on the side that lays to the wrist. This leaves enough space for storing electronics. Since the parts are only connected by hinges, routing the necessary wires between the segments needs to be considered, e.g. by leaving small holes right next to the hinges.



(a) CAD view



(b) Hinge Detail

Figure 5: Multi-part design

Since the manufacturing possibilities at the Human-Computer Interaction (HCI) group don't allow for slow hinges (as the inspiration by Muschke suggests), a different solution for keeping the bracelet closed while worn on the wrist needed to be considered. A magnet clasp with small neodymium magnets was tested, but attaching them turned out to be more difficult than expected. When attached to the inside of the segment tips, the magnetic force was too weak for reliably keeping the bracelet together. Mounting the magnets on the outside of the segments resulted in mounting difficulties, since the magnetic force of neodymium magnets is very strong and frequently resulted in torn glue layers.

When an electrophoretic display was considered to be part of the bracelet, the segmented approach became undesired, since in addition to aforementioned difficulties in designing and manufacturing, the usable surface space was relatively small when it came to hosting a single big component like a display. This issue led to further research into various other manufacturing techniques and potential materials for bracelet prototypes, and eventually led to cast silicone.

4.2.3 *Silicone Bracelet*

After some consideration on a flexible e-ink display, a prototype bracelet made of silicone was considered. The molds used in the casting process were designed and printed just as the bracelets presented in the previous sections. However, designing a mold was significantly more difficult.

As with the previous 3d printed prototypes, the bracelet positive was designed first. The flexibility of silicone allows for some features that weren't practical when implemented in rigid material, for example cavities for electronic components. This resulted in overall more complex bracelet concepts. In addition, closing mechanisms like magnets had to be considered in this design stage.

When the CAD process for the positive is finished, the mold is designed by adding surrounding geometries to the model and applying boolean difference operations. If reusability is desired for individual mold components or the whole mold, the geometry and constellation of the mold parts needs to be considered and the characteristics of the printed material have to be taken into consideration. For example, tunnels in the resulting silicone bracelet are not possible if the mold should be usable more than once.

Another important aspect of mold design is planning the casting process. Some types of silicone rubber are more viscous than others, and the mold design needs to make sure that the liquid rubber reaches all corners and delicate parts well. The distribution inside the mold can be supported by applying gentle vibration during the cure process, but this assumes that the liquid silicone is already distributed into most regions of the mold.

The first silicone design was a simple strap with an open pocket for the long electrophoretic display which was considered as a feature of the bracelet at that time in the design process. The closing mechanism relied on molded neodymium magnets for which some cavities were featured in the bracelet design.

Mold design turned out a little tricky but finally succeeded. A two-piece mold was printed which needed a little post-processing to fit properly together. To ensure a proper fit when closed, small tongues and grooves were added too the mold parts. The mold's inside was coated with black spray paint to make it a little smoother, but this didn't work as intended so the painting was omitted in future mold making processes. The filling holes featured for the silicone were too small, and the mixture was more viscous than expected, so the first cast failed and resulted in two small end pieces and nothing in between.

For following casts, one part of the mold was filled with silicone and closed afterwards; this turned out significantly better. The orientation in which the mold is placed during the dry period is also relevant, as air bubbles would float towards the "top" of the mold, leading to instabilities when oriented inappropriately. After the liquid silicone's cure period, getting the cast out of its mold was no problem.

This first rubber bracelet design involved a magnet clasp, but it was infeasible to reliably attach magnets to silicone with anything but silicone itself and they would likely jump out of place and snap together if placed too close to each other in the design. Apart from that, the silicone felt much more appealing on the skin and was perceived less obstructive while worn on the wrist. The possibility to attach components by placing them in pockets or cavities resulted in thinner bracelets in general.

Two casts were made, one with each of the available silicone mixtures. The softer one was slightly too soft and had a repulsive color. In addition, the cavity rims were too short, so the display would jump out of place almost instantly when bending the bracelet. With this experience and the positive impressions of the material itself, more silicone designs were produced and manufactured.

4.2.4 *One-Piece Silicone Bracelet*

The next design was a ring-like silicone bracelet in one piece, so the issue with closing mechanisms was no concern. The mold for this prototype consisted of three pieces, an outer and an inner ring as well as a bottom plate to properly align those rings. Due to the very rigid characteristic of the printed material, the molds could not be recovered after the cast and had to be destroyed in order to retrieve the bracelet. In order to prevent unnecessary waste of material, the wall strength was reduced to 2.5 mm which turned out to be strong enough to survive assembly and the casting process. The advantage of using one-time molds was the possibility to add tunnels to the design. This was used in some models, especially for the display area to hide cables or bulky connectors.

Those molds were harder to fill with material than the previous one. Especially the Sorta-Clear silicone was too viscous to fill the complete mold, resulting in broken casts. The green MoldStar silicone turned out to work quite well, but sometimes it would leak through little gaps between the rings and the base plate, so a thorough assembly was desired more than ever. Remaining gaps were closed with hot glue in later iterations. In addition, a simple self-made vibrating plate helped filling all parts of the mold and reduced the number of air bubbles in the cast.

The first one-piece bracelet designs included an electrophoretic display and featured wide cavity rims to hold said display in place. First wearing tests resulted in success. When the display was omitted later on, the cavities stayed to hold a capacitive touch surface in place. After the final hardware configuration took form, another cavity for the electronics was added to the design. Since the hardware was thicker than the bracelet, the spot to house it needed to be enlarged in thickness. The board was eventually placed on the inner wrist and connects to the touch element with a short flat wire cable along the bracelet's surface. It is powered by a micro USB cable attached to a port on the lower rim of the bracelet. A status LED is visible through the silicone on the upper rim above the electronics.

4.3 TEENSY DEVELOPMENT BOARD

The core part of the bracelet's electronics is the Teensy USB development board, which is built around a MK20DX256 32 bit ARM Cortex-M4 Processor running on 72 MHz clock speed [35]. The Teensy can be programmed using the popular Arduino IDE and thus can profit from the great amount of existing libraries and code examples for the Arduino family.

Like all Cortex processors, the Teensy's chip has direct capability to process capacitive touch input. It also features an I^2C module for communication with other components (see also section 4.5), several serial communication ports and an integrated programmer to enable flashing via USB. However, the processor includes no floating point computation unit, which slows down calculations that feature floating point numbers.

4.4 TOUCH SLIDER

The major input interface of the bracelet is a touch surface which consists of seven cut copper foil segments that are placed in a zigzag pattern. This capacitive strip can either be used as an array of seven individual buttons or as a single large surface that can detect swiping gestures or a primitive variant of multi-touch interaction.

Capacitive touch is based on the concept of a parallel plate capacitor. The copper foil segments act as a capacitor's plate, the human hand touching it represents the other plate. Since the capacitance is proportional to the area of the plates, larger segments lead to increased sensitivity. The definition of a plate capacitor's capacitance C is defined as follows:

$$C = \frac{A\epsilon}{d}$$

Where A represents the area of the plates, ϵ is a material constant of the dielectric between the plates and d is the distance between the plates. This shows that the capacitance changes more when the finger is close to the sensor and less when it's farther away.

Touch sliders can be compared to a potentiometer and return an analog value for the finger's position on the slider [6]. The usual layout of a touch slider features several neighboring electrodes, in the context of this thesis, a slider with seven electrodes was designed and built, since the Teensy has a restricted number of capacitive sensing enables pins from which some were already in use for other components, since almost all pins of the chip serve multiple functions.

A zigzagged pattern of the slider's electrodes supports microstepping, i.e. considering the logical combinations of multiple electrodes pressed at the same time to achieve a higher resolution in position detection [6].

4.5 3D ACCELEROMETER

Another interaction interface of the bracelet is a MMA8652FC three-axis digital accelerometer [10]. In a micromachined device like the one used in this thesis, a seismic mass is attached to delicate "springs" in order to measure the acceleration. Both components are usually made of silicone. When an acceleration occurs, the electric capacitance between the seismic mass and the fixed frame changes and the occurring acceleration can be derived accordingly. Note that one such mechanism is only able to detect acceleration along a single axis. Since the MMA8652FC is a three-axis accelerometer, it contains on mass-spring-system for each of the three axis.

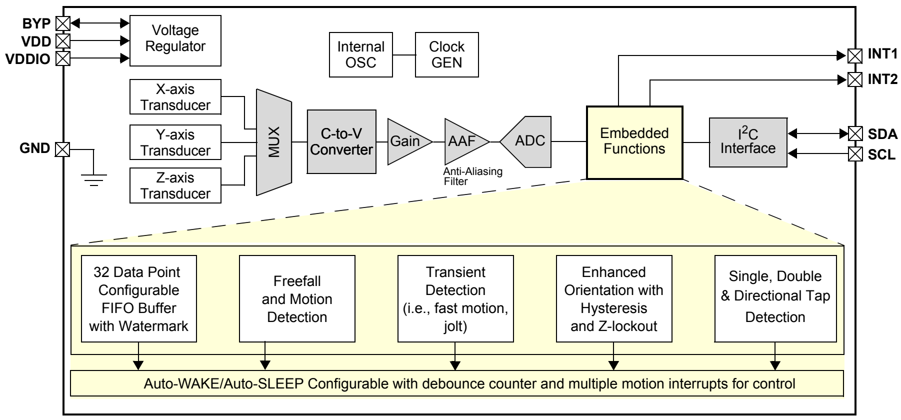


Figure 6: Block diagram of the MMA8652FC digital accelerometer. [10]

In addition to this basic accelerometer functionality, the MMA8652 also features a range of embedded microprocessing functions, as the component's block diagram (fig. **??**) depicts. Out of those functions, the "Single, Double & Directional Tap Detection" was used intensively. This module can detect tap interactions, which can be visualized as spikes in the acceleration along one specific axis. The detection process can be configured in detail to adjust factors like the tap intensity or the duration between the two taps of a double tap. When tap detection is enabled, a single register's content indicates if such a tap has occurred. The detailed configuration used for single and double tap recognition in context of this thesis is discussed in section 6.7.

In order to communicate with the bracelet's processor, the accelerometer implements the I²C communication protocol which was developed in 1982 by Philips Semiconductor (now NXP Semiconductors) and became public domain in 2006 when the underlying patent expired. By using two lines for a clock signal (SCL) and data transmission (SDA) respectively, data can be interchanged in a master-slave system at varying bit rates. Figure **??** illustrates this process. To initiate communication, the Teensy incorporates the role of a master device and begins a transmission by changing the SDA value from high to low while SCL is on high. Now the bus is considered busy until the master sends the stop condition and the actual data request and transmission can take place.

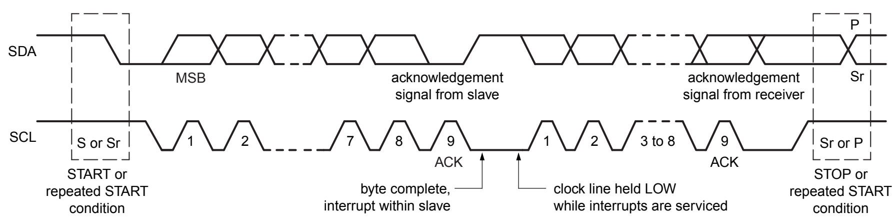


Figure 7: Data transfer on the I²C bus. [25]

To request register data from the accelerometer, the master writes the address of the desired register on SDA. Note that no further synchronization between master and slave device is needed, since the slave adapts the clock signal on SCL which is determined by the master device. After sending the register's address, the Teensy indicates the end of the transmission without sending a Stop signal. The accelerometer acknowledges the received data and proceeds to send the requested register's contents, and finishes by terminating the connection with a Stop command, which is generated by switching SDA from low to high while SCL is on high. Both lines now stay high due to connected pull-up resistors. If the next register's content is requested, the process starts again with a Start signal transmitted by the master [25].

4.6 VISUAL FEEDBACK

To provide visual feedback to the user, an RGB LED is incorporated in the bracelet. Since most changes are reflected by the light source with no perceptible delay, the LED serves mainly as an indicator for the recognition of specific interactions like the double tap detection.

5

INTERACTIVE LIGHT SOURCE

The previously mentioned light source for the scenario is built around a 12 V powered RGB LED strip, which can be purchased for home lighting or car decoration. It is approximately XX cm long and features XX LEDs on each XX cm long segment. The original controller box which included an Infrared (IR) receiver for a remote was removed and replaced by an Arduino Uno [3] with a custom shield.

The shield features a Roving Networks RN42 Bluetooth module [29] and some transistors for upscaling the voltage from 3.5 V to 12 V. The whole setup is powered by a standard 12 V power supply and encased in a spherical lamp shade made from translucent glass in order to diffuse the spotted impression of the LED strip (cf. figure **??**).

Figure 8: The interactive light source, encased.

Communication with the bracelet takes place via Bluetooth. The RN42 module implements the Bluetooth Serial Port Profile (SPP) and can be accessed easily with the `SoftwareSerial` module from the Arduino standard library. The lamp is configured as a Bluetooth slave, the bracelet acts as the master device. For command transmission, a serial connection is established over the Bluetooth link between the devices. Once it is set up, the master transmits a nine-digit string composed from three values from 000 to 255. Note that all values must have three digits, leading zeros must not be omitted. These values represent the intensities of the red, green, and blue channel respectively. The algorithm on the Arduino feeds them into the platform's analog outputs which feature Pulse-width Modulation (PWM). These output signals are scaled as mentioned before and so the light color changes. A simple fading algorithm prevents disruptive flashing while switching colors.

It turned out that the RGB LEDs use a lot of current, a test with an adjustable power supply yielded that the lights become much brighter with increased current. The power supply was limited to XX Ampere and there was no saturation in bightness indicated at this level. However, the standard power supply mentioned above serves only XX A which results in less brightness. A way to improve this would be to shorten the LED strip by cutting off several segments.

6 INTERACTION

This chapter describes the various interaction levels with the bracelet in detail and illustrates the respective algorithms.

6.1 PAIRING THE BRACELET WITH A LIGHT SOURCE

The bracelet's Bluetooth device automatically searches for nearby devices and connects to a range of matching IDs without any interaction or confirmation by the user. This can pose a security risk to spoofing a lamp device, but since no sensitive data is handled by either participant, the impact would rather be an annoying disturbance than a serious threat to privacy.

6.2 SWITCHING THE LIGHT SOURCE ON AND OFF

Since turning the lights in a specific room on or off is a frequently used interaction, it should require few complexity in terms of cognitive as well as physical workload, i.e. a simple, easily memorable way of interaction is much desired. For those reasons, simply covering the touch surface with the whole hand and holding for a few seconds will result in switching the lights on or off, dependent on the current state.

6.3 ADJUSTING THE LIGHT SOURCE'S BRIGHTNESS

Apart from switching the lights on or off, a change in brightness is the second most desired interaction in the smart lighting scenario. For example, the incentive of watching TV in the living room benefits from a dimmed light setting. However, if the user fails to locate the remote control for the television, a quick dim interaction towards brighter light facilitates the search for the missing remote control. After the item is found, the brightness level of the room's lighting can easily be dimmed back to the desired setting.

The dimming interaction is different from the other use cases, since there exists a physical solution for this task in form of dim knobs for wall outlets. Usually, those wall dimmers are rotary knobs connected to a potentiometer which dims the light source by increasing the electric resistance. Hence, the interaction of turning a knob for dimming the light level is an association for many people.

The intention was to preserve this association to make the interaction with the bracelet more intuitively. The dimming interaction is a combination of twisting the wrist while covering the bracelet's touch surface, imitating the interaction with the wall dimmer. A counter-clockwise movement reduces the brightness, while a clockwise twist increases the brightness.

6.4 CHANGING THE LIGHTING MOOD

In addition to the brightness, users should be able to change the *mood* of the current

6.5 PRECISE TOUCH INPUT FOR COLOR CHANGE

6.6 TEMPLATE PICKING BY GESTURE RECOGNITION

The most casual form of interaction with the bracelet is by drawing gestures in the air to trigger basic operations, e.g. switching a light source on or off. These gestures are recorded by the bracelet's accelerometer and processed using the "3$ Gesture Recognizer" [18], an extension of the popular "1$ Recognizer" by Wobbrock et. al [38]. The algorithm is explained in detail in the following paragraphs.

After a gesture is recorded, it is resampled to a fixed number of points. If the gesture was drawn quickly, it would have less samples and thus less points compared to a slowly drawn gesture. In order to be able to compare these two gestures, a resampling is performed before further processing. The length of the gesture path M is calculated and an increment size I derived by dividing M by $(N/1)$, where N is the desired number of samples. The gesture path is stepped trough and after each distance I, a new point is inserted using linear interpolation (cf. listing 1).

Listing 1: Resampling of a points path into N evenly spaced points

```
def resample(points, N):
        I = path_length(points) / (N - 1)
        D = 0
        newpoints = points[:1]
        for i in range(1, len(points)):
                dist = distance(points[i-1], points[i])
                if(D + dist) > I:
                        q = points[i-1] + ((I - D)/dist) * (
                            points[i] - points[i-1])
                        newpoints.append(q)
                        D = distance(q, points[i])
                else:
                        D = D + dist
        if(D + dist) > increment: #append last point manually
                q = newpoints[-1] + ((increment - D) / dist) * (
                    points[i] - newpoints[-1])
                newpoints.append(q)
```

```
        return newpoints
```

In the next step, the gesture is prepared for matching against the templates by rotating it to a specific position. The so called *indicative angle* θ between the gesture's centroid C and its first point is calculated using the normalized scalar product of the centroid and the first point's position vectors. Afterwards the gesture is rotated so that θ is at $0°$. Wobbrock et. al. do this by using the inverse tangens function, however this is not possible in 3D space. Hence, *Rodrigues' rotation formula* (named after French mathematician Olinde Rodrigues) was implemented instead. This efficient algorithm for rotating a vector in $\mathbb{R}^3$ takes the rotation angle θ as well as the axis unit vector k as input, and calculates the following formula:

$$v_{rot} = v\cos\theta + (k \times v)\sin\theta + k(k \cdot v)(1 - \cos\theta). \text{ [16]}$$

After applying this function on the gesture, a good starting point is created for the actual recognizing part of the algorithm. Listing 2 illustrates the rotation procedure.

Listing 2: Rotation of points so that their indicative angle is at 0°

```
def rotate_to_zero(points):
        c = centroid(points)
        theta = acos(points[0] * c / (|points[0]| * |c|)
        newpoints = rotate_by(points, -theta)
        return newpoints
```

In order to harmonize gestures of different sizes, the points are scaled to a reference cube with edge length of 100 units and translated so that the respective centroid C is on the origin (cf. listing 3).

Listing 3: Scaling to reference cube and translation to origin

```
def scale_and_translate(points, size): # size=100
        B = Bounding_Box(points)
        newpoints = []
        for p in points:
                q = Point()
                q.x = (p.x * (size / B.width)) - c.x
                q.y = (p.y * (size / B.height)) - c.y
                q.z = (p.z * (size / B.depth)) - c.z
                newpoints.append(q)
        return newpoints
```

After these harmonization and preparation steps, the gestures are matched against the prerecorded templates (cf. listing 4). The aforementioned steps are applied to templates as well as to newly recorded gestures, the following algorithms only apply to unrecognized gestures.

Listing 4: Matching candidate gesture against every template

```
def recognize(points, templates, rescale_size):
        theta_min = -180
        theta_max = 180
        theta_delta = 2

        best = float("inf")
        for t in templates:
                dist = distance_at_best_angle(points, t,
                    theta_min, theta_max, theta_delta)
                if dist < best:
                        best = dist
                        t_best = t
        return t_best
```

The candidate is compared to all stored templates using the average Mean Square Error (MSE) as a scoring metric. The optimal position between the candidate gesture and a stored template is possibly offset by a certain rotation, so the optimal combination of angles between the two gestures needs to be determined. Since rotations are costly in terms od computation time, the candidate gesture should be aligned to the template in as few tries as possible. Hence, as proposed by [18], a Golden Section Search (GSS) is used to find the optimal angles α, β, and γ for rotation around the three axis of the coordinate system.

The GSS algorithm was invented by US-American statistician Jack Kiefer in 1953 and is conceptualized to find the minimum value x^* of a unimodal function $f(x)$ in a given interval $[a, b]$ [14]. Two function points $f(x_1)$ and $f(x_2)$ with

$$x_1 = a + (1 - \phi)(b - a) \qquad\qquad x_2 = a + \phi(b - a)$$

are calculated and compared, where $\phi = 2/(1 + \sqrt{5})$ denotes the Golden Section constant. If $f(x_1) < f(x_2)$, the interval for the next iteration becomes $[a, x_1]$ and the new test points are calculated as described above with respect to the new interval borders. Note that the value $f(x_1)$ can be reused [7]. If $f(x_1) > f(x_2)$, the interval for the next iteration step changes to $[x_2, b]$ respectively. If the function points differ not more than a given x_Δ, the current minimum will be accepted as the algorithm's result.

In the context of the implemented gesture recognizer, the function to be minimized is the distance between a gesture candidate and a certain template at the best angle. Since the recognizer works with three-dimensional data, there is not a single angle for rotation, but one fir each coordinate axis. Hence, the GSS needs to be adapted for three dimensions.

As listing 5 illustrates, the three-dimensional approach is similar to the one-dimensional algorithm. Each of the variables α, β, γ has its own search interval and a pair of calculated values. This leads to eight function points to be evaluated in each iteration, denoted in the code by variables `f1` to `f8`. The minimum of these values is calculated and the intervals adjusted accordingly. Note that in every case one function value from the previous step can be carried over, for example when `f1` is the minimum, its value is used for the new `f8`.

Listing 5: Three-dimensional Golden Section Search for finding the best angle between a candidate gesture and a template

```
def distance_at_best_angle(p, t, theta_min, theta_max,
    theta_delta):
        phi = 0.5 * (-1 + math.sqrt(5))
        a_min = theta_min
        a_max = theta_max
        b_min = theta_min
        b_max = theta_max
        g_min = theta_min
        g_max = theta_max

        x1 = phi * a_min + (1 - phi) * a_max
        x2 = (1 - phi) * a_min + phi * a_max
        y1 = phi * b_min + (1 - phi) * b_max
        y2 = (1 - phi) * b_min + phi * b_max
        z1 = phi * g_min + (1 - phi) * g_max
        z2 = (1 - phi) * g_min + phi * g_max

        f1 = distance_at_angle(p, t, x1, y1, z1)
        f2 = distance_at_angle(p, t, x1, y1, z2)
        f3 = distance_at_angle(p, t, x1, y2, z1)
        f4 = distance_at_angle(p, t, x1, y2, z2)
        f5 = distance_at_angle(p, t, x2, y1, z1)
        f6 = distance_at_angle(p, t, x2, y1, z2)
        f7 = distance_at_angle(p, t, x2, y2, z1)
        f8 = distance_at_angle(p, t, x2, y2, z2)

        while (|a_max - a_min| > theta_delta) or (|b_max - b_min|
             > theta_delta) or (|g_max - g_min| > theta_delta):
                min_f = min(f1, f2, f3, f4, f5, f6, f7, f8)
                if min_f == f1: #x1, y1, z1
                        a_max = x2
                        x2 = x1
                        x1 = phi * a_min + (1 - phi) * a_max
                        b_max = y2
                        y2 = y1
                        y1 = phi * b_min + (1 - phi) * b_max
                        g_max = z2
                        z2 = z1
                        z1 = phi * g_min + (1 - phi) * g_max
                        f8 = f1
                        f1 = distance_at_angle(p, t, x1, y1, z1)
                        ...
                        f7 = distance_at_angle(p, t, x2, y2, z1)
                elif min_f == f2: #x1, y1, z2
                        ...
```

```
            else: #x2, y2, z2
                    ...
        return min(f1, f2, f3, f4, f5, f6, f7, f8)
```

To determine the distance between a gesture and a certain template at given angles α, β, and γ, the gesture is rotated using the Rodrigues rotation formula discussed in the context of the indicative angle above. The rotation formula needs an axis and an angle for executing the rotation, but the GSS works with three angles. Therefore, the rotation matrix formed by the angles needs to be transformed into a Euler axis/angle pair for use in the implemented rotation formula. The authors of [18] state the following rotation matrix in their reference implementation [17]:

$$A = \begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{pmatrix}$$

The Euler rotation angle θ is composed by the matrix's diagonal elements.

$$\begin{aligned} \theta &= \arccos(\frac{1}{2}(A_{11} + A_{22} + A_{33} - 1)) \\ &= \arccos(\frac{1}{2}(\cos\alpha\cos\beta + \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma + \cos\beta\cos\gamma - 1)) \end{aligned}$$

The rotation axis vector e can be derived from A and θ as follows:

$$\begin{aligned} e_1 &= \frac{A_{32} - A_{23}}{2\sin\theta} \\ &= \frac{\sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma - \cos\beta\sin\gamma}{2\sin\theta} \\ e_2 &= \frac{A_{13} - A_{31}}{2\sin\theta} \\ &= \frac{-\sin\beta - \cos\alpha\sin\beta\cos\gamma - \sin\alpha\sin\gamma}{2\sin\theta} \\ e_3 &= \frac{A_{21} - A_{12}}{2\sin\theta} \\ &= \frac{\cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma - \sin\alpha\cos\beta}{2\sin\theta} \end{aligned}$$

An so, the angles determined in algorithm 5 lead to an axis/angle rotation which is performed on the candidate gesture.

Once the best template for a candidate gesture is found, the corresponding index in the stored template list is returned and the recognized gesture triggers certain changes in the light source's color or brightness. Note that this gesture recognition algorithm is only used for recognition of the preset gestures formulated in section XX.

REGISTER NAME	PARAMETER	VALUE
PULSE_THSZ	Tap Detection Threshold	100g
PULSE_TMLT	Interval between Start and End Pulse	6.25ms
PULSE_LTCY	Ignore Interval after Detection	25ms
PULSE_WIND	Maximum Double Tap Interval	500ms

Table 2: Single- and double tap detection configuration for the MMA8652FC digital accelerometer

6.7 PRESETS AND CONFIGURATION

Gesture recording and recognition is triggered by a gentle double tap on the bracelet. This small but focused activation reduces unwanted triggering of the gesture recognition process, e.g. while gesturing heavily, and thus reducing false positives. The tap detection functionality is a built-in feature of the bracelet's accelerometer, configuration parameters for this process are listed in table 2.

In order to be recognized as a tap interaction, the initial impulse needs to be at least XX g in intensity. When calibrated like this, jerky movements like suddenly raising the hand at a high speed are correctly not recognized as a tap. However since the threshold is that high, the activation tap needs to be executed directly on the hardware which is located on the inner wrist.

The PULSE_TMLT register configures the maximum time interval between the impulse exceeding the threshold on the Z axis and falling back under said threshold. If the mentioned interval lasts at most 6.25ms, the interaction is considered as a tap.

After a tap is detected, all impulses in the following 25ms are ignored by the detection mechanism. This prevents bouncing effects and detecting multiple taps in a singe tap movement.

The MMA8652FC accelerometer is able to distinguish between single and double taps. The last configuration register listed in table 2 is a parameter for double tap detection. It specifies the maximum time interval between two double taps and is set to 500ms, the same time interval as the Windows default between two mouse clicks of a double click [22].

7 EVALUATION

7.1 STUDY DESIGN

7.2 RESULTS

7.3 DISCUSSION

8 CONCLUSION AND FUTURE WORK

BIBLIOGRAPHY

[1] 3D Systems, Inc. ProJet x60 Series - Professional 3D Printers, 2014. URL http://www.3dsystems.com/about-us.

[2] 3D Systems, Inc. About 3D Systems, 2014. URL http://www.3dsystems.com/about-us.

[3] Arduino. Arduino Uno, 2014. URL http://arduino.cc/en/Main/ArduinoBoardUno.

[4] Daniel Ashbrook and Thad Starner. MAGIC: A Motion Gesture Design Tool. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2159–2168. ACM, 2010.

[5] W Buxton. Integrating the periphery and context: A new taxonomy of telematics. In *Proceedings of graphics interface*, volume 95, pages 239–246, 1995.

[6] Oscar Camacho and Eduardo Viramontes. Designing Touch Sensing Electrodes. *Freescale Semiconductor Application Note*, Document Number: AN3863, Rev. 4:1–27, 2011.

[7] Yen-Ching Chang. N-dimension golden section search: Its variants and limitations. In *Biomedical Engineering and Informatics, 2009. BMEI'09. 2nd International Conference on*, pages 1–6. IEEE, 2009.

[8] Fitbit Inc. Fitbit flex, 2014. URL https://www.fitbit.com/flex.

[9] Freecad. An open source parametric 3D CAD modeler, 2014. URL http://freecadweb.org.

[10] Freescale Semiconductor. Xtrinsic mma8652fc 3-axis, 12-bit digital accelerometer. *Freescale Semiconductor Data Sheet*, Document Number: MMA8652FC, Rev. 3, June 2014. URL http://cache.freescale.com/files/sensors/doc/data_sheet/MMA8652FC.pdf.

[11] Google. Android wear, 2014. URL http://www.android.com/wear/.

[12] IFTTT, Inc. About IFTTT, 2014. URL https://ifttt.com/wtf.

[13] Marije Kanis, Niall Winters, Stefan Agamanolis, Anna Gavin, and Cian Cullinan. Toward Wearable Social Networking with iBand. In *CHI'05 Extended Abstracts on Human Factors in Computing Systems*, pages 1521–1524. ACM, 2005.

[14] Jack Kiefer. Sequential minimax search for a maximum. *Proceedings of the American Mathematical Society*, 4(3):502–506, 1953.

[15] Jungsoo Kim, Jiasheng He, Kent Lyons, and Thad Starner. The gesture watch: A wireless contact-free gesture based wrist interface. In *Wearable Computers, 2007 11th IEEE International Symposium on*, pages 15–22. IEEE, 2007.

[16] Don Koks. *Explorations in Mathematical Physics*. Springer Science+Business Media, 2006.

[17] Sven Kratz. Google code: $3 gesture recognizer, June 2010. URL https://code.google.com/p/three-dollar-gesture-recognizer.

[18] Sven Kratz and Michael Rohs. A $3 gesture recognizer: simple gesture recognition for devices equipped with 3D acceleration sensors. In *Proceedings of the 15th international conference on Intelligent user interfaces*, pages 341–344. ACM, 2010.

[19] Seungyon Claire Lee, Bohao Li, and Thad Starner. AirTouch: Synchronizing in-air hand gesture and on-body tactile feedback to augment mobile gesture interaction. In *Wearable Computers (ISWC), 2011 15th Annual International Symposium on*, pages 3–10. IEEE, 2011.

[20] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. uWave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, 5(6):657–675, 2009.

[21] Kent Lyons, David Nguyen, Daniel Ashbrook, and Sean White. Facet: A Multi-Segment Wrist Worn System. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, pages 123–130. ACM, 2012.

[22] Microsoft Corporation. Windows Dev Center - Desktop: GetDoubleClickTime function, 2014. URL http://msdn.microsoft.com/en-us/library/windows/desktop/ms646258.aspx.

[23] Daniel Muschke. Amico Bracelet, June 2012. URL http://grabcad.com/library/amico-bracelet--9.

[24] Tao Ni and Patrick Baudisch. Disappearing Mobile Devices. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology*, pages 101–110. ACM, 2009.

[25] NXP Semiconductors. I^2c-bus specification and user manual. *User Manual*, Document Number UM10204, Rev. 6, April 2014. URL http://www.nxp.com/documents/user_manual/UM10204.pdf.

[26] Pebble Technology, Corp. Pebble watch, 2014. URL https://getpebble.com/pebble.

[27] Henning Pohl and Roderick Murray Smith. Focused and casual interactions: Allowing users to vary their level of engagement. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2223–2232. ACM, 2013.

[28] Razer Inc. Razer nabu, 2014. URL http://nabu.razerzone.com/.

[29] Roving Networks. RN-42 Class 2 Bluetooth Module, April 2013. URL http://ww1.microchip.com/downloads/en/DeviceDoc/rn-42-ds-v2.32r.pdf.

[30] Samsung Electronics. Samsung galaxy gear, 2014. URL http://www.samsung.com/global/microsite/galaxynote3-gear/spec.html.

[31] Susanne Seitinger, Daniel S Perry, and William J Mitchell. Urban pixels: Painting the City with Light. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 839–848. ACM, 2009.

[32] Susanne Seitinger, Daniel M. Taub, and Alex S. Taylor. Light Bodies: Exploring Interactions with Responsive Lights. In *Proceedings of the Conference on Tangible, Embedded and Embodied Interfaces*, 2010.

[33] Smooth-On, Inc. Mold StarÂ® 15, 16 and 30, 2014. URL http://www.smooth-on.com/tb/files/MOLD_STAR_15_16_30_TB.pdf.

[34] Smooth-On, Inc. SORTA-clear series, 2014. URL http://www.smooth-on.com/tb/files/SORTA_CLEAR_SERIES_TB.pdf.

[35] Paul Stoffregen and Robin Coon. Teensy usb development board, 2014. URL https://www.pjrc.com/teensy/index.html.

[36] Aneesh P Tarun, Byron Lahey, Audrey Girouard, Winslow Burleson, and Roel Vertegaal. Snaplet: Using Body Shape to Inform Function in Mobile Flexible Display Devices. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 329–334. ACM, 2011.

[37] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L Littman. Practical Trigger-Action Programming in the Smart Home. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems*, pages 803–812. ACM, 2014.

[38] Jacob O Wobbrock, Andrew D Wilson, and Yang Li. Gestures without libraries, toolkits or training: a $1 recognizer for user

interface prototypes. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 159–168. ACM, 2007.

EIDESSTATTLICHE ERKLÄRUNG

Hiermit versichere ich, die vorliegende Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die wörtlich oder inhaltlich aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Hannover, December 2014

Karoline Busse