

Martynas Dabravalskis

Conductive 3D printing for touch interfaces

MSci Hons Computer Science

19th March 2021

## Declaration

I certify that the material contained in this dissertation is my own work and does not contain unreferenced or unacknowledged material. I also warrant that the above statement applies to the implementation of the project and all associated documentation. Regarding the electronically submitted version of this submitted work, I consent to this being stored electronically and copied for assessment purposes, including the School's use of plagiarism detection systems in order to check the integrity of assessed work.

I agree to my dissertation being placed in the public domain, with my name explicitly included as the author of the work.

Date: 19th March 2021

Signed:

A handwritten signature in black ink, consisting of a series of loops and a long horizontal stroke at the end.

## Abstract

This project aims to explore the usage of conductive 3D printer filament in combination with regular filament to create interactive 3D printed objects with integrated capacitive touch areas. Even though 3D printing has become increasingly popular over the last few years, the objects created are still very simple static objects. As such, the usage of conductive filament should greatly increase the interactivity of 3D printed products.

The project involves the design and printing of an interactive, cube-shaped touch interface via the use of conductive 3D printing. Insights are provided into the process of creating such an object, as well as both hardware and software limitations. A study is conducted examining the gesture detection capabilities of the manufactured interface.

## Table of Contents

<b>1. Introduction .....</b>	<b>6</b>
<b>1.1 Aims and Objectives .....</b>	<b>6</b>
<b>1.2 Overview .....</b>	<b>7</b>
<b>2. Background.....</b>	<b>8</b>
<b>2.1 Tangible user interfaces.....</b>	<b>8</b>
<b>2.2 Touch Sensors .....</b>	<b>9</b>
<b>2.3 Printed Touch Sensors.....</b>	<b>9</b>
<b>3. Design.....</b>	<b>12</b>
<b>3.1 Hardware .....</b>	<b>12</b>
<b>3.1.1 3D Model .....</b>	<b>12</b>
<b>3.1.2 Capacitive Touch Panels.....</b>	<b>13</b>
<b>3.2 Software .....</b>	<b>14</b>
<b>3.3 Requirements .....</b>	<b>15</b>
<b>3.3.1 Functional requirements.....</b>	<b>15</b>
<b>3.3.2 Non-Functional requirements.....</b>	<b>15</b>
<b>4. Implementation.....</b>	<b>16</b>
<b>4.1 Components and tools.....</b>	<b>16</b>
<b>4.1.1 Electronic components .....</b>	<b>16</b>
<b>4.1.2 Structural components.....</b>	<b>16</b>
<b>4.1.3 Plastic Filament.....</b>	<b>16</b>
<b>4.1.4 3D printer .....</b>	<b>17</b>
<b>4.2 Software used .....</b>	<b>17</b>
<b>4.2.1 3D design software .....</b>	<b>17</b>
<b>4.2.2 3D printing software .....</b>	<b>17</b>
<b>4.2.3 Microcontroller software .....</b>	<b>17</b>
<b>4.2.4 Client software .....</b>	<b>17</b>
<b>4.2.5 Gesture detection software.....</b>	<b>17</b>
<b>4.3 Cube body .....</b>	<b>18</b>
<b>4.3.1 First prototype.....</b>	<b>18</b>
<b>4.3.2 Final prototype.....</b>	<b>18</b>
<b>4.4 Cube side panels.....</b>	<b>19</b>
<b>4.4.1 First prototype.....</b>	<b>19</b>
<b>4.4.2 Intermediate prototype .....</b>	<b>20</b>
<b>4.4.3 Final prototype.....</b>	<b>21</b>
<b>4.5 Final cube model.....</b>	<b>21</b>
<b>4.6 Electric circuit.....</b>	<b>23</b>
<b>4.7 Software .....</b>	<b>24</b>
<b>4.7.1 Arduino Software.....</b>	<b>24</b>
<b>4.7.2 Gesture detection software (Creating and training the model) .....</b>	<b>25</b>

4.7.3 Client software – Bluetooth / Gesture Detection .....	27
4.7.4 Client software – UI .....	28
5. Testing and Evaluation .....	30
5.1 Sensor Testing.....	30
5.2 Gesture Detection Study .....	30
5.2.1 Results .....	31
5.3 Static Program Analysis .....	32
5.4 Battery Testing.....	33
5.5 Reusability Testing .....	33
5.6 UI Testing.....	33
5.7 Fulfilment of Requirements.....	33
5.7.1 Functional requirements.....	33
5.7.2 Non-Functional requirements.....	34
6. Conclusion.....	35
6.1 Review of aims .....	35
6.1.1 Designing and printing an interactive interface .....	35
6.1.2 Writing software for the interpretation of touch gestures .....	35
6.1.3 Exploring the use cases of the printed object .....	35
6.1.4 Simplifying the process of creating 3D printed touch interfaces/gesture detection software.....	35
6.2 Technical and Personal Challenges.....	35
6.3 Future Work .....	36
6.4 Closing Remarks.....	37
7. References .....	38
8. Appendix.....	40
8.1 Project proposal.....	40
8.2 Research information sheet .....	44
8.3 Research Participant Consent Form .....	47

# 1. Introduction

3D printing has become increasingly popular over the last few years. As the technology continues to mature and 3D printer production and operating costs continue to decline it is estimated that the 3D printing market size will continue to grow at a compound annual growth rate (CAGR) of over 14% for the next seven years (Berman, 2012).

3D printing enables small quantities of customised goods to be produced at relatively low costs (Grand View Research, Inc, 2020). This allows for rapid production of customizable products without the need for expensive and time-consuming development of full production lines.

However, most products are limited by the materials used to create them. As 3D printing usually involves a single type of plastic or other material used to print the product (Kaufui V. Wong, 2012), the products created are passive and do not support any type of interactivity by themselves.

One way of adding interactivity support to 3D printed objects is by adding touch support. Touch sensors are widely used in smart displays, smart glasses, and other hand-held devices.

The previously static model could then be used as a tangible user interface (TUI), where the user can interact with other systems through the physical environment by touching the object. TUIs take advantage of the natural ability of humans to understand and manipulate physical forms (Ishii, 2008), tangible user interfaces are especially useful in Augmented Reality applications, where a digital world is manipulated via special input devices (Dimitrios Chamzas, 2020). While spatial manipulation of the physical objects is usually used for these applications, incorporating touch sensors would enable touch input along the physical object's surface, thus allowing for direct interaction with the virtual object itself.

Currently, by far the most popular technology for providing touch support is capacitive touch sensing (Mackey, 2011). With traditional 3D printing, capacitive touch sensing would normally be introduced by external means, such as adding copper touch plates to the object after it has been printed (C. Shemelya, 2013). However, the new availability of conductive printing materials, as well as the availability of multi-material printing technology should allow for the printing of interactive products. A typical plastic filament, such as an ABS or PLA plastic, can be used in combination with conductive 3D filament to print objects with integrated conductive touch plates, as well as internal plastic wiring to enable easy access to the sensor plates.

The project aims to explore the possibilities of an interactive object printed using the method mentioned above. The project involves creating a fully interactive 3D printed cube with integrated touch plates. The shape of such an interactive object was chosen to be a cube, as cubic shapes allow for a wide variety of interaction possibilities (Kevin Lefeuvre, 2018). The paper describes the steps taken in designing and manufacturing a fully working prototype.

To explore the interactivity of the designed product, the object is to be designed to allow for various touch gestures to be registered via the use of a microcontroller board with custom software. The paper explores how different touch sequences or gestures can be used as control inputs. A demo client application is developed, which allows the user to map different gestures of the cube to different computer keyboard key events.

## 1.1 Aims and Objectives

The aim of the project is to explore the usage of conductive 3D filament in creating a fully interactive touch interface as well as explore the touch gestures that can be achieved by utilizing the integrated capacitive touch sensors and custom gesture detection software.

To achieve these aims, the following objectives will be involved:

- Designing and printing the interactive interface – A custom 3D model is to be created and printed for this project. Separate capacitive touch areas will need to be included in the model, as well as the necessary wiring that would allow for the capacitive surfaces to be connected to an external microcontroller board through a single access point on the cube model. A few different-sized models, as well as models with different sensor placement, will need to be created to optimize the object.
- Writing software for the interpretation of touch gestures – A software will need to be written that would allow for the microcontroller to interpret the touch inputs received from the sensors. The software will be made to understand the various touch gestures and translate them into direct commands to a computer.
- Exploring the use cases of the printed object – Touch gestures will be implemented and explored. A demo software will be designed, which will allow the user to record custom touch gestures and map them to different computer keyboard events.
- Simplifying the process of creating 3D printed touch interfaces/gesture detection software – The development process will be described in detail to provide others with references on creating 3D printer touch interfaces. The software will be designed so that it could be reused in similar projects.

## 1.2 Overview

**Chapter 2:** Background Research. Discusses the related research in printed touch sensors, interactive devices, and gesture detection. Provides insight into the reason for some of the design decisions made in this project.

**Chapter 3:** Design. This chapter discusses the design decisions taken on both the hardware and software sides of the project and discusses the initial. It also provides insight into the initial ideas behind the project.

**Chapter 4:** Implementation. This chapter provides detail on the steps taken to implement the project, as well as provides information on some of the iterations of the device.

**Chapter 5:** Testing and Evaluation. This chapter provides information on the testing that was performed and the results of those tests.

**Chapter 6:** Conclusion. This chapter reviews the aims set forth at the start of the project, discusses the challenges faced throughout the project, and provides suggestions on future work in this area of research.

## 2. Background

### 2.1 Tangible user interfaces

A tangible user interface (TUI) has first been defined by Hiroshi Ishii in 1997 as an interface, that can “augment the real physical world by coupling digital information to everyday physical objects and environment” (Hiroshi Ishii, 1997).

Kenneth P. Fishkin notes, that this definition is rather vague and can encompass many different devices (Fishkin, 2004). His paper proposes classifying Tangible User interface devices by their Embodiment and Metaphor.

Embodiment represents “How closely tied is the input focus to the output focus”. The paper presents four levels of this characteristic: Full, nearby, environmental, and distant. Each subsequent level is less embodied, where full embodiment is when the output device is the input device, and distant embodiment, where the output can be an external screen.

The metaphor represents the similarity of the virtual effect to the physical input effect. The paper categorizes TUIs by metaphor with four levels: None, Noun, Verb, and Full. Noun, in this case, is when “an analogy is made to the physical shape/look/sound of object(s) in the system”. In the case of Verb, the analogy is made to the act that is being performed. Many tangible user interfaces can be metaphorized as both Noun and Verb.

Several benefits are offered by tangible user interfaces. They encourage two-handed interactions and can take advantage of our natural spatial reasoning skills to control systems (Fitzmaurice, 1995). Additionally, They can help understand virtual model representations by taking advantage of the natural ability of humans to understand and manipulate physical forms (Ishii, 2008). An example is Illuminating clay, which can help analyse landscapes by projecting their virtual representation on clay (Piper, 2002).

Although any physical shapes can be used for interactive tangible devices, rectangular shapes are especially common in all types of TUIs. Examples include the AlgoBlock system, which is a set of physical blocks that can be connected manually to form a program (Hideyuki Suzuki, 1995), and Tangible Programming Bricks, which are general-purpose research tool that can be repeatably programmed (McNerney, 2000). The programming is performed by inserting parameter cards and stacking a Bricks on top of another.

These two examples mainly focus on moving the shapes themselves. This paper will focus on improving the tangible interaction experience by interacting with the object’s surface, by performing different touch actions.

The decision of choosing a cubic shape as an interactive prototype was also supported by a paper that highlights the advantages of cubic shapes for designing interactive devices (Kevin Lefevre, 2018). The paper discusses various properties of cubic shapes that make them a prevalent choice in designing interactive interfaces.

According to the paper, a cube has a “natural affordance for manipulation”. Due to their 90° angles, they accurately represent all 3 spatial axes, unlike other shapes like spheres, which do not have clearly defined sides. As such, cubes are widely used as input devices where the object is manually manipulated with one or two hands to produce desired input. Additional sensors can also allow for rotation and acceleration to be reliably manipulated in all axes.

The paper also mentions that a cube is one of the easiest shapes to model and construct. It also provides a detailed list of example devices, where different properties of a cube are used as methods of interaction.



Adding to the previously mentioned research, this paper (Jean-Baptiste de la Rivière, 2008) explores the touch interaction techniques of cubic-shaped interfaces in more detail. The touch surfaces on the cube allowed for the manipulation of virtual 3D objects by implementing common multi-touch gestures for translation, scale, and rotation of the virtual object. The 3D multi-touch interface proposed in this paper could be created with the use of printed conductive touch sensors.

## 2.2 Touch Sensors

Touch sensors, also known as tactile sensors, are used to detect and record physical touch. They are small, simple, low-cost electronic sensors that replace the traditional mechanical switches used in the past (Seed, 2020).

Capacitive sensing is by far the most used touch sensing technology today (Du, 2016) (Mackey, 2011). It is used in many devices, such as mobile phones, smartwatches, mobile phones, tablets, TVs.

Capacitive touch sensors detect the changes in capacitance between a charged electrically conductive plate and a finger or other conductor of electricity. There are two main types of capacitive sensing technology: self-capacitive and mutual-capacitive sensors. A self-capacitive system measures changes in capacitance with respect to the ground, where the sensed object is the ground. Mutual capacitive uses two plates, where the object to be detected ‘disrupt’ the capacitance between the plates, instead of using the object as a ground plate (All About Circuits, 2016).

One of the papers I found (Grosse-Puppenthal, 2017) provides a thorough review of capacitive sensing in the context of human-computer interaction.

As shown in Figure 2.1, a plethora of natural capacitances exist between the people, devices, and objects in the environment. By measuring these values it is possible to infer relative position, motion, and more—supporting a multitude of interaction techniques and applications (Grosse-Puppenthal, 2017). As such, it has become a common part of many interactive devices.

The paper mentions several advantages of capacitive sensing over other technologies. Sensors are easy to produce and can be run with simple electronics that require no moving parts. Additionally, the sensors can work both at very close distances and very short distances. This is especially important, as other technologies, such as resistive sensing, only work when the sensors are touched directly.

In the context of touch interaction, this may allow for more touch techniques to be recognized, as it allows for sensing of the distance between the sensor and a human hand/finger. In terms of 3D printing, this would also allow for the touch plates to be hidden behind a non-conductive layer, which means that the printed sensing surface does not necessarily have to be exposed.

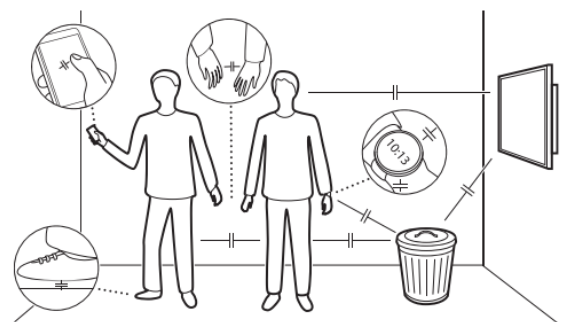


Figure 2.1 Natural capacitance between people, their devices, and conductive objects in the environment (Grosse-Puppenthal, 2017).

## 2.3 Printed Touch Sensors

Most interactive devices and the capacitive touch sensors themselves are manufactured via subtractive manufacturing, where a part is created by removing parts of an initial material

(Autodesk, 2018). For example, capacitive touch electrodes can be created on printed circuit boards by etching away copper from a plated surface (Digi-Key, 2018).

On the other hand, additive manufacturing creates objects by depositing materials, often in layers, to form the desired objects. Additive manufacturing may offer several advantages over the subtractive process.

A review of Additive manufacturing (Kaufui V. Wong, 2012) describes and highlights the benefits of several additive manufacturing processes.

To begin with, Additive processes do not require custom moulds or tools for specific parts, so 3D models of various shapes can be directly produced by a machine. This significantly lowers the cost and increases the speed of prototype or low-quantity production.

Additionally, additive manufacturing is affordable and can be used by hobbyists. Machines that cost as little as 200£ or less are now widely available (Technology Outlet, 2019). This, coupled with available and cheap materials, and accessible 3D design software, have made 3D printers, which use this process, widely used by hobbyists (Berman, 2012).

Additive manufacturing can be divided into several different processes, but by far the most popular one is Fused Deposition Modelling. According to statistics (Grand View Research, Inc, 2020), most of the 3D printers sold use FDM.

As the review of Additive manufacturing describes, it is a process that uses thin filaments of plastic, which are melted to form layers of the desired object. It typically uses plastic materials, such as acrylonitrile butadiene styrene (ABS) or polylactic acid (PLA).

The commonly used plastics, by themselves, are not conductive and as such cannot print conductive objects, as is required for capacitive sensors. As such, the only way to add touch sensing surfaces is externally embedding conductive into the model after it has been printed. This technique was demonstrated in one paper (C. Shemelya, 2013), where copper wire and copper wire meshes were added to allow for touch input to be perceived.

However, composite printer filaments, which can have conductive properties, are getting increasingly more popular. Multiple electrically conductive plastic filaments are now available in the market (All3DP, 2021), which could be used to print conductive capacitive sensor electrodes.

As conductive plastic is a relatively new material, the possibilities of directly printing conductive touch electrodes have not been thoroughly explored.

One of the few existing papers provides a lot of technical details about designing and 3D printing of capacitive touch sensors for interactive objects (Martin Schmitz, 2015). The paper reports on how to incorporate 3D printed touch surfaces in static 3d printed objects via multi-material 3D printing. The paper reports on the implementation of such sensor surfaces, as well as their technical limitations.

To show the viability of using plastic filament for accurate capacitive sensing, the paper evaluated the performance of 3D printed touch sensors with respect to their conductivity and dimension. The paper provides guidelines for electrode sizes and their maximum resistance for robust touch sensing, as well as the maximum size of an overlay (non-conductive layer) that could be used to keep the electrodes hidden.

To keep the SNR (Signal to noise ratio) below an acceptable 5:1, the authors recommend an electrode resistance lower than 30 k $\Omega$  and an overlay layer no thicker than 10mm.

The paper will prove as a baseline for many of the design and implementation choices made in this project.

However, even though this paper does present several examples and the applications of the 3d printed objects with integrated capacitive touch sensors, it does not explore the interaction possibilities in more detail. The paper does not expand on the implementation of complex touch gestures and how they can be used to interact with the object.

### 3. Design

Cube, a simple geometric shape was chosen as the body of the touch interface. It is a shape that is easy to manipulate in space and is comfortable to use. A cube can be moved in 3D space, picked up, and touched. The cube can be paired with an application, where these interaction methods can be used as user input.

There are several potential applications for a touch-enabled cube.

One of the use cases is as a tangible user interface for Augmented Reality. A user may use a smartphone application to automatically pair with the interactive object. The application would then, by using the camera of the smartphone, provide an enhanced version of the cube, with which the user could interact by using the physical object.

The cube could also be used as a simple controller. Touch gestures and other types of interaction with the cube could be used as input for video games and other types of software.

As a proof of concept, a simple use case will be demonstrated in this project. Custom software will be created that converts different touch gestures into keyboard events. These keyboard events can then be easily used with various software. For example, the user will be able to use touch gestures to pause, fast-forward and rewind videos by converting touch gestures into YouTube keyboard shortcuts.

The following subchapters discuss the design decisions taken on both the hardware of the model and the software that provides functionality to the cube.

#### 3.1 Hardware

##### 3.1.1 3D Model

The object needs to be small enough to be comfortable to use with one hand, yet large enough to be able to perform gestures with repeatable success.

The first idea involved creating a dual extruded<sup>1</sup> model.

Dual extrusion printing is preferable, as it makes it possible to integrate a completely different plastic into the model during printing, in this case, conductive filament for the touch surfaces.

A quick prototype model was designed (see figure 3.1). This was the starting point of the design process and showcased the possibility of integrating conductive plastic into 3D printed bodies, with plastic 'wires' thick enough to be of reasonable electrical resistance. The model contained 5 touch surfaces on different sides of the cube and was 5cm<sup>3</sup> in size.

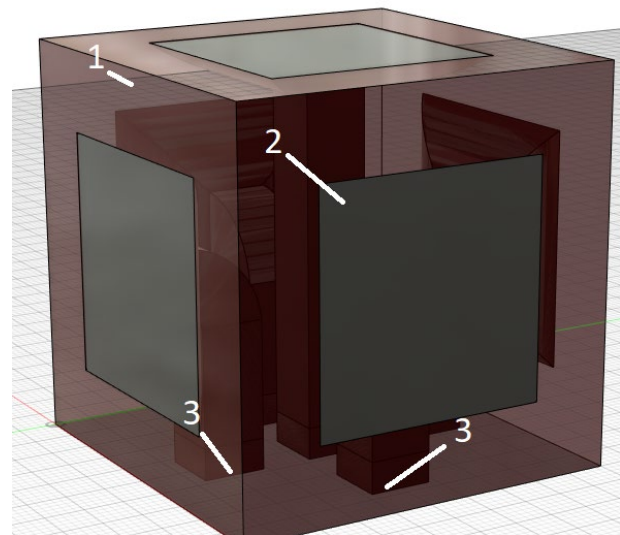


Figure 3.1 | 1 – Cube body, 2 – Touch surfaces, 3 – Wire connectors

<sup>1</sup> Dual extrusion – A printing process which allows for an object to be printed with two different materials at the same time

However, several flaws of this type of model were immediately discovered. As the cube itself is not hollow, all electronics need to be separate from the cube, and wireless operation of the touch interface is impossible.

Furthermore, due to the COVID-19 pandemic, access to 3D printers capable of dual extrusion was impossible. As such, the conductive and non-conductive parts of the model would have to be printed separately. This posed new challenges, which will be further discussed in the implementation section of the report.

Due to the reasons mentioned above, it was decided to create a hollow cube, where the electronic parts could be placed internally. The cube would contain a microcontroller with Bluetooth capabilities and would be powered with a battery, enabling wireless operation. It will consist of six attachable panels with capacitive touch areas on each side with the conductive touch areas placed from inside the side panels. They should allow for all sides of the cube to be interactable and maintain easy access to the electronics for charging and maintenance.


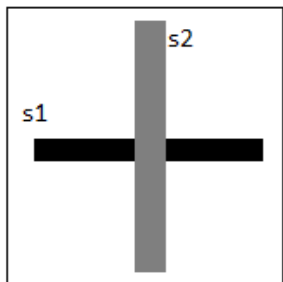
### 3.1.2 Capacitive Touch Panels

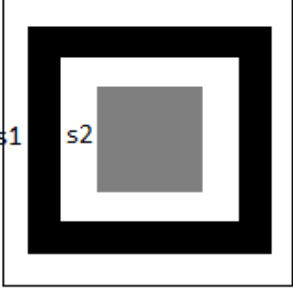
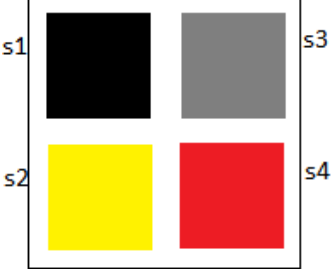
Self-capacitive sensors were chosen for this project. Touch in this type of sensor is sensed when a change in capacitance is detected between a conductive human finger, which is in this case the ground electrode, and the 3D printed electrically charged conductive plate.

The possible touch gestures that may be performed on the cube are mostly defined by the number of capacitive touch sensors and their configuration. For example, a single sensor on each side of the cube can only detect whether the side is touched but adding more sensors on each side may allow for more complex motions, such as a swipe.

In order to decide on the best sensor configuration, a comparison table was created:

Table 3.1 Touch sensor configurations

Configuration	Two touch surfaces on opposite sides	
Advantages	<ul style="list-style-type: none"> <li>Swiping left - &gt;right and vice versa</li> <li>Scale, zooming (Starting at the sides, move fingers closer to the center and vice versa)</li> </ul>	
Disadvantages	<ul style="list-style-type: none"> <li>Detection might not work well depending on how close the surfaces are to each other</li> <li>Can only detect swipes in one axis.</li> <li>If the touch areas are too far apart, touching the middle area might not register</li> </ul>	
Configuration	Two touch surfaces in a cross formation	
Advantages	<ul style="list-style-type: none"> <li>Swipe detection in all axes may be possible</li> <li>Scale, zooming (Starting at the sides, move fingers closer to the center and vice versa)</li> </ul>	
Disadvantages	<ul style="list-style-type: none"> <li>Corner touches may not be registered.</li> <li>Detection of most gesture may require relying on analog sensor readings</li> </ul>	
Configuration	A middle touch surface with an outer touch surface	
Advantages	<ul style="list-style-type: none"> <li>Swiping in all directions</li> <li>Possible scale, zoom gestures.</li> </ul>	

	<ul style="list-style-type: none"> <li>• Touch should be detected on every part of the panel</li> </ul>	
Disadvantages	<ul style="list-style-type: none"> <li>• Cannot easily distinguish swipe directions (might be possible with analog readings from other panels)</li> </ul>	
<b>Configuration</b>	<b>Four touch surfaces in a square formation</b>	
Advantages	<ul style="list-style-type: none"> <li>• Swiping in all directions</li> <li>• Scale, zoom gestures.</li> <li>• Circular motions with distinguishable direction</li> </ul>	
Disadvantages	<ul style="list-style-type: none"> <li>• Would require 2 capacitive touch sensor breakout boards</li> <li>• Would require double the amount of wires/ increased complexity</li> </ul>	

A few different configurations were tested in the implementation part of the project, but a configuration of a middle touch surface with an outer touch surface was chosen in the end due to its simplicity and the satisfactory number of possible touch gestures that can be performed.

### 3.2 Software

The software that needs to be created for this project can be divided into three parts:

- Microcontroller software – This software sends sensor data to the client.
- Gesture detection software – This software creates and trains a machine learning model on the recorded data.
- Client software – This software uses the sensor data to detect touch gestures/events. It also provides a functional UI.

The microcontroller software will run on the device itself and is responsible for the setup of sensors, reading, and transmission of sensor data. The data will be transmitted via Bluetooth or similar wireless technology. A wireless approach was selected, as it allows for the cube to be tangible from all sides.

Gesture detection and Client software will both run on a Windows computer. The client software's main purpose is to provide UI. It will allow the users to connect to the device, train, detect gestures, and map selected gestures to specified virtual keyboard keys. The gesture detection itself will be done on the Client software, and the gesture detection software was called to create/train an AI model based on the data provided by the user.

### 3.3 Requirements

#### 3.3.1 Functional requirements

Table 3.2 Functional requirements

ID	Description
1	The touch interface shall be able to detect touch input
1.1	The touch interface shall be able to detect touch input over a 1mm distance
2	The cube shall have at least one sensor on each side
3	The cube shall be able to sense acceleration
4	The cube shall be able to sense its rotation
5	The cube shall be able to transmit sensor data via Bluetooth
6	Touch gestures shall be detected
6.1	Simple gestures shall be detected
6.2	Complex gestures shall be detected
7.	The user shall be able to set custom touch gestures
8.	The GUI shall be able to display the actions performed on the interface on a virtual 3D model.

#### 3.3.2 Non-Functional requirements

Table 3.3 Non-Functional requirements

ID	Description
1.	The touch interface shall be able to detect touch input in all human finger capacitance conditions
2.	Touch gestures shall be detected with 70% accuracy
2.1.	Touch gestures shall be detected with 80% accuracy when there is no noise from other sensors (When the interface is static)
2.2.	Touch gestures shall be detected with 60% accuracy when there is noise from other sensors (When the object is being held in hands)
3.	The battery should last at least 7 hours when the interface is in use.
4.	The Software shall support the addition of more than 12 sensors.
5.	The Software shall be able to support custom 3D models.
6.	The user shall be able to set custom touch gestures



## 4. Implementation

This chapter outlines core implementation elements that were built for this project. The project can be divided into two discrete sections: the interactive 3D printed model and the software. The interactive object will be detecting touch input and sending the data over Bluetooth. The software will be interpreting the touch inputs and translating them into gestures. An iterative rapid prototyping approach was used, as it enables rapid refinement and testing of each prototype.

### 4.1 Components and tools

#### 4.1.1 Electronic components

- **Adafruit Feather nRF52840 Express** was chosen as the microcontroller. The nRF52840 SoC (System on a chip) this microcontroller uses is faster and has more memory than most Arduino boards, while the board itself is cheaper than most of the original Arduino boards. It has built-in battery charging and Bluetooth low energy, which removes the need for separate breakout boards for these functions. It is also compatible with Arduino IDE.
- **Adafruit MPR121 12-Key Capacitive Touch Sensor Breakout board** was used to detect touch. It allows for the connection of 12 separate touch plates and connects to a microcontroller via an I2C 2-wire bus. This was the only choice available in the market, as most breakout boards use chips that only allow for one capacitive touch plate.
- **A small 3.7V 110mAh Lithium polymer battery** was used to power the electronics. It measures 3x13x33mm, and as such it is small enough to fit in the cube. The capacity of the battery should be enough considering the relatively small power usage of the system.

When Idle (not connected via Bluetooth), the battery should last over 20 000 hours (4.5 $\mu$ A at 3V). When transmitting sensor data, the battery should last around 11h (+4dBm, ~10mA at 3V) Charging should take less than an hour with a charging current of 200mA.

#### 4.1.2 Structural components

- **M2x10mm screws**
- **M2 washers**
- **M2 nuts**

#### 4.1.3 Plastic Filament

- **U3 CONDUCTIVE ABS 2M filament by U3PRINT** was used as the conductive filament for this project. It was the only reasonably priced conductive filament available in the region. It is specified to have a conductance of 4.64\*10<sup>2</sup> Ohms/cm.
- **White and black ABS filaments** were chosen. As the conductive filament uses ABS plastic, it was decided to use ABS filaments to print the structure of the cube, as different types of plastic shrink by different amounts when cooled, which results in differing dimensions of objects.



#### 4.1.4 3D printer

A **Prusa i3 Rework type DIY 3D printer** was used to print the plastic components. Other 3D printers were unavailable due to COVID-19 related restrictions.

### 4.2 Software used

#### 4.2.1 3D design software

While Tinkercad software was chosen at first to design the 3D model, due to the simple structure of the model, Fusion360 design software was used instead, as it is still easy to use and has advanced tools which were required for the model, such as chamfer and fillet.

#### 4.2.2 3D printing software

Ultimaker Cura was used to slice the stl model files. It is easy to use and supports a variety of advanced settings.

Printrun was used to communicate with the 3D printer, as that is the most popular 3D printing host suite used with RepRap(ref) printers.

#### 4.2.3 Microcontroller software

Arduino IDE was used to program the microcontroller. It was chosen as it is easy to use and has libraries for implementing Bluetooth functionality, capacitive touch, accelerometer/gyroscope sensors.

#### 4.2.4 Client software

Unity was used for designing the client program. It has 3D graphics support and makes it simple to create user interfaces. It is mainly used to create games, which is one of the use cases for interactive physical objects. Gesture detection was also implemented inside unity.

#### 4.2.5 Gesture detection software

The gesture detection training software was written in Python. Tensorflow was used as it is a powerful, yet simple to use machine learning platform. Python 3.7.9 with Tensorflow 1.15.5 was used. Newer versions could not be used due to compatibility issues.

### 4.3 Cube body

The model went through many iterations until the final one was achieved. In the following sections, different changes made throughout these iterations will be described.

#### 4.3.1 First prototype

The cube was divided into two parts (see Figure 4.1). The first one (no. 1) is the outer cube part which houses the inner cube (no. 2) as well as slots for the conductive sides (no. 3). The inner cube houses the main electronic components: The microcontroller (no. 4), the MPR121 capacitive touch sensor breakout board (no. 5) as well as other electronic components such as the battery and other sensors.

The outer cube is 10cm<sup>3</sup>, while the inner cube is 6cm<sup>3</sup>. The inner cube took around 5 hours to print with a layer height of 0.2 mm.

It was calculated that the outer cube would take more than 12 hours to print, and as such, I decided that it would have to be removed. I also decided that a 10cm<sup>3</sup> cube would be too large as it would not fit in one hand comfortably.

The design would go through many iterations to address these and other issues that would appear after seeing the printed product.

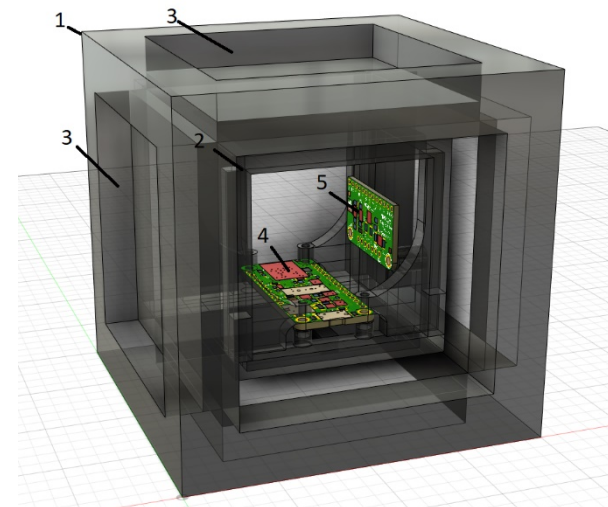


Figure 4.1 First Prototype model

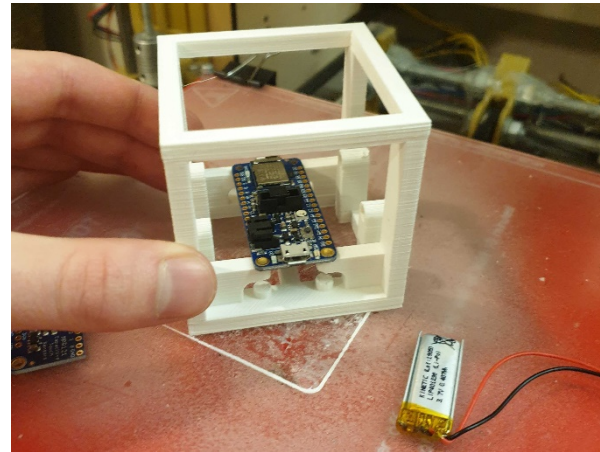


Figure 1.2 Printed inner cube of the first prototype

#### 4.3.2 Final prototype

The final design (see Figure 4.3) consists of the main structure, snap-fit slots, and holders for the circuit boards.

The outer cube was removed and the size of the inner cube was increased, which will now house the slots for side plates, to 7cm<sup>3</sup>.

Snap-fit slots were also added. A 50° tapered angle was chosen for the snap-fit, as the trial prints showed that such an angle would hold the side plates in place securely while still making it easy to remove them.

The electronic components are held in place by screws that screw into the holders. These holders also provide some structural support.

M2 screws were used to screw the electronic components in place, while the battery was secured with double-sided tape.

Jumper wires were used to connect the electronic components, though some wires had to be soldered due to space limitations. Glue was used in certain places where the jumper wires were not held securely.

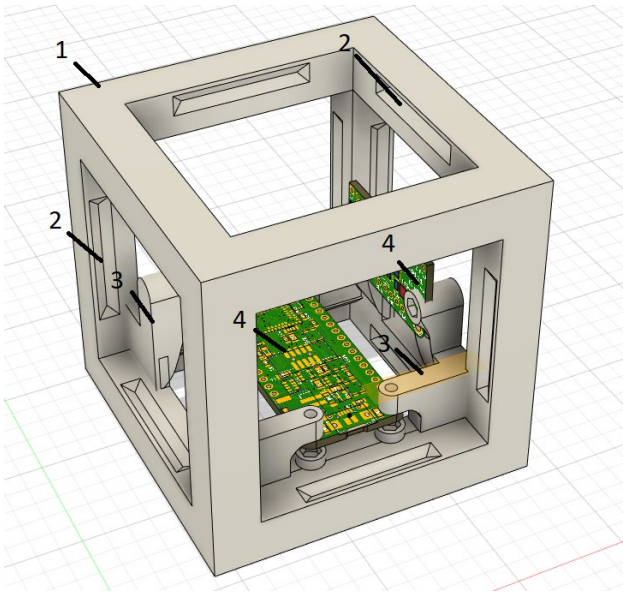


Figure 4.3 Final prototype model | 1 - main structure, 2 – Snap-fit slots, 3 – Screw slots, 4 – Electronic boards

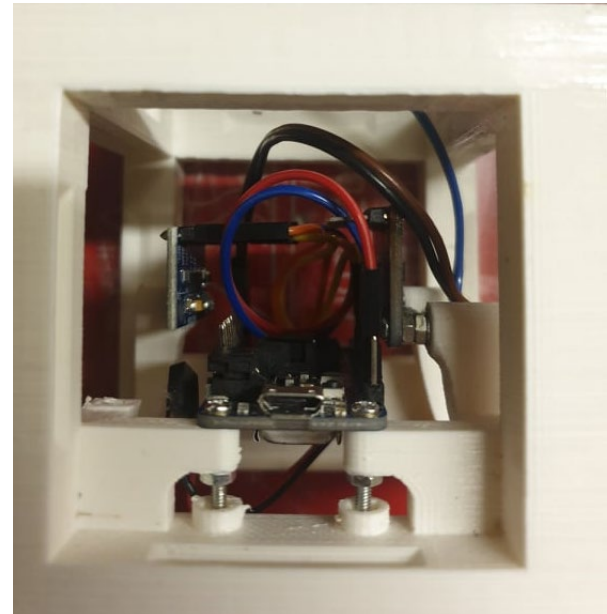


Figure 4.4 Printed final prototype, with the assembled electronics

## 4.4 Cube side panels

Like the cube model itself, the side panels also went through many iterations until the final one was achieved. Each iteration provided some improvements over the previous one. Print settings were also tuned throughout the process. In the following sections, different changes made throughout these iterations will be described.

### 4.4.1 First prototype

The first design (see Figure 4.5) of the side contained the non-conductive body (grey) with 4 conductive touch surfaces (red) that get inserted into the panel from the outside.

The touch surfaces are exposed to the users and thus allow for direct touch sensing, as such increasing the accuracy of the touch sensors.

However, due to the printer only being able to print with one material at once, the conductive plates have to be printed separately. This causes the touch plates to fit imperfectly into the side panels, which causes bumps when swiping and makes the whole cube less aesthetically pleasing.

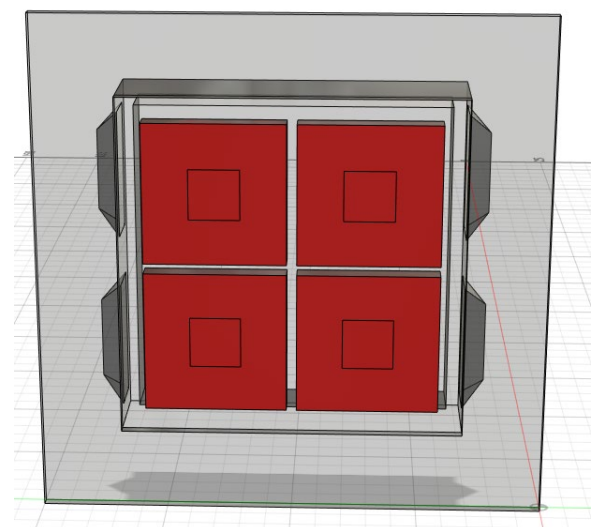


Figure 4.5 Model of the first panel prototype

It was also decided that, while 4 touch surfaces allow for more complex gestures, it would require two capacitive touch breakout boards and the increased number of wires would be very difficult if not impossible to fit inside the cube.

The snap-fit joints were also badly designed and would allow movement of the panel in the y axis relative to the cube body (up and down, where there are no joints). This also only allowed for the plates to be mounted in two directions. The snap-fit joints themselves were also too thick and stiff, which makes them very difficult to snap in place.

#### 4.4.2 Intermediate prototype

The side plate (see Figure 4.6) consists of the non-conductive body (grey) and conductive touch plates (red).

The side plate now only contains two touch areas, the configuration of which is shown in the picture.

The snap-fit joints are now smaller and more flexible, which makes them easy to snap in place. They are also now located on each side of the plate to decrease their movement when inserted. However, the snap-fit joints are now too brittle and can break easily when inserting.

The conductive are inserted from inside of the side panel, which fixed some of the issues discussed before.

However, it was discovered that placing the conductive surfaces beneath a layer of non-conductive plastic caused the touch sensitivity to drop dramatically and touches would not register even when placing your whole hand on the surface of the plate.

Measurements showed that the resistance of the conductive plastic was too great and the non-conductive gap too large (see Figure 4.7). According to research, a resistance lower than  $30\text{k}\Omega$  is needed for reliable touch sensing (Martin Schmitz, 2015).

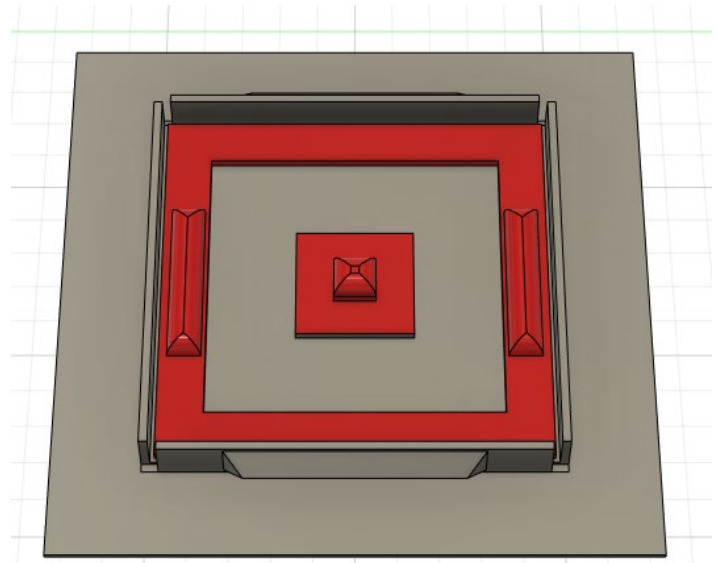


Figure 4.6 Model of the intermediate panel prototype

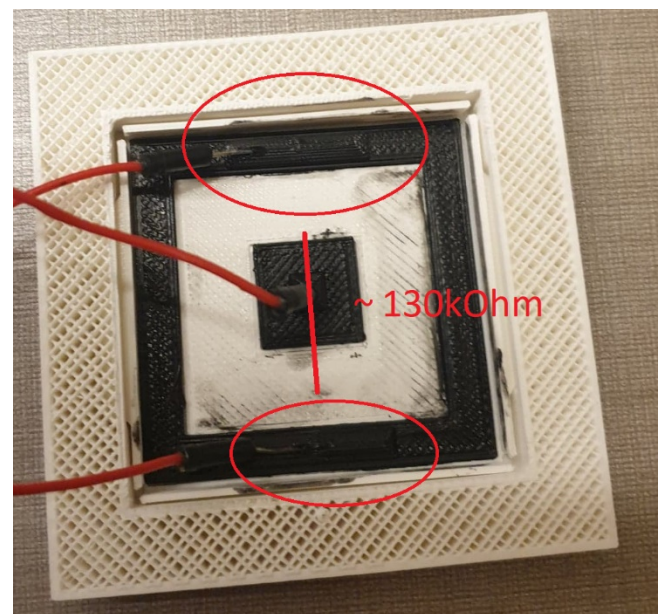


Figure 4.7 Resistance of the printed touch plates



#### 4.4.3 Final prototype

To address the existing issues, multiple changes were made.

The conductive areas in the final prototype (see Figure 4.8) now cover most of the plate area, which makes touch detection possible even when touching the outer parts of the panels.

The conductive areas are now wider and thicker. They were also printed at higher temperatures and a higher extrusion rate, which increases layer adhesion and the density of the conductive parts respectively. These changes decreased the resistance dramatically, as shown in Figure 4.9.

The non-conductive body was also made thinner to reduce the distance between the conductive touch plates and the user's fingers. While this made the whole side plate flexible, it was a necessary step to increase touch sensitivity.

The snap-fits were made thicker to increase their strength while some unnecessary obstructive parts were removed.

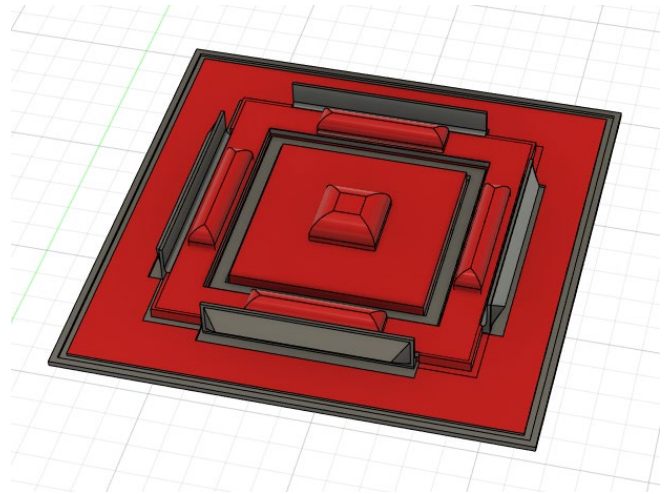


Figure 4.8 Model of the final side panel prototype

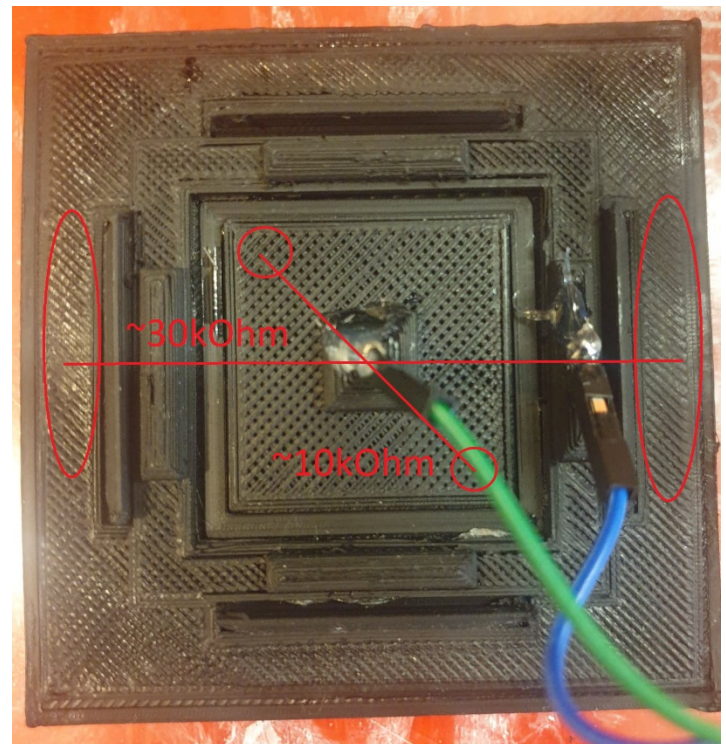


Figure 4.9 Resistance of touch plates of the final prototype

#### 4.5 Final cube model

The cube structure took ~10h to print at 250°C print temperature, 90°C build plate temperature at 50mm/s infill, 20mm/s wall, 40mm/s support speeds.

0.2mm layer height was used as the cube structure does not need to be very detailed, as it will be mostly hidden by the side plates. The low speeds were used due to printer limitations. Hairspray was used to increase first layer adhesion, as ABS plastic tends to warp and can easily pop off the printing surface. Structural support was generated for angles greater than 80°.

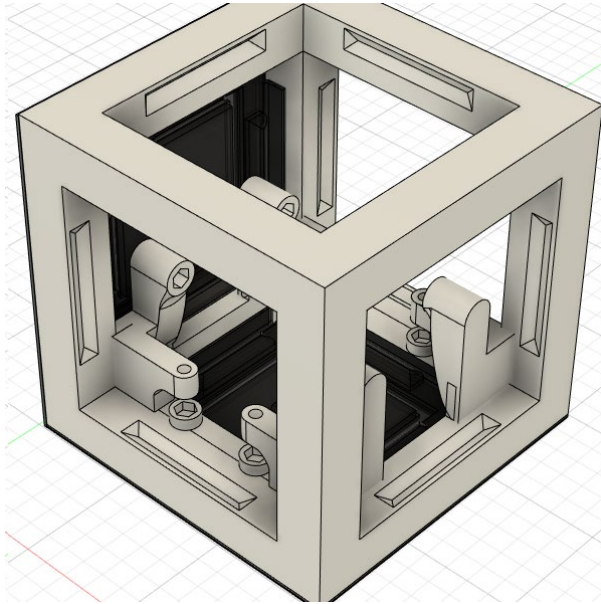
3 cube side panels were printed in a single run, each run taking 4h for the conductive parts, and 3h for the non-conductive parts. 0.1mm layer height was used, as greater precision and detail were needed. The non-conductive panel parts were also printed with retraction enabled to avoid the dripping of plastic.

In total, the whole model took ~38h to print.

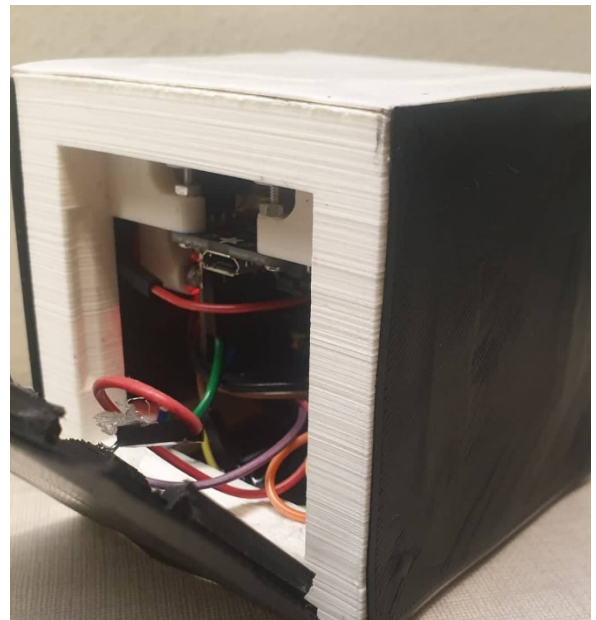
The dimensions of the final cube are 7x7x7cm, and the final weight, including all the parts, is 104g.

The conductive areas were glued to the non-conductive plastic with acetone (acetone dissolves ABS plastic). However, it was noted that using acetone to glue plastics of different colours causes the dyes to leech into each other. Hot glue also proved to be a viable method.

The wires were inserted into the conductive areas by soldering. Though optional, hot glue was also used to keep the wires securely in place so that they do not get easily removed by pulling.



*Figure 4.10 The final cube model*



*Figure 4.11 Printed and assembled final cube model*

## 4.6 Electric circuit

The electric circuit is very basic. All the electronic components (as described in section 4.1.1) are connected as shown in the circuit diagram below:

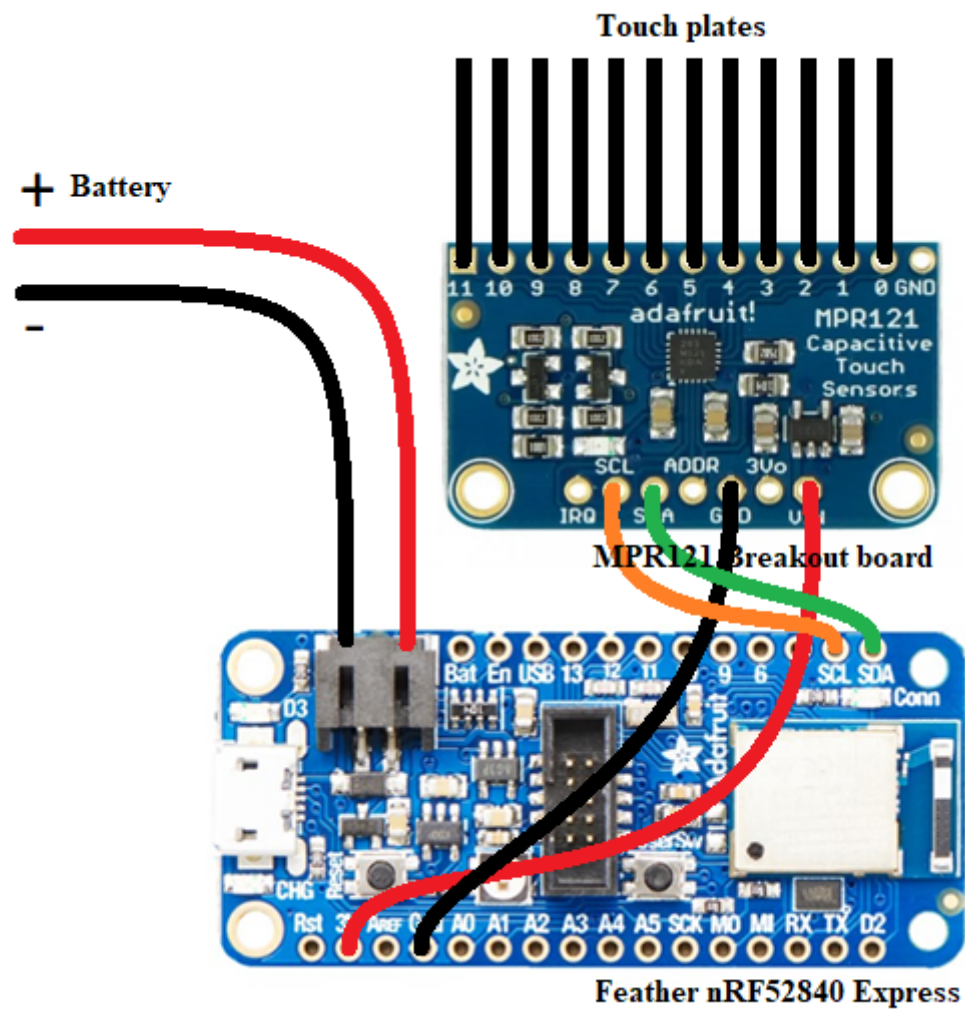


Figure 4.12 Electric circuit diagram

## 4.7 Software

### 4.7.1 Arduino Software

The Arduino implementation was quite straightforward since sensor data reading and Bluetooth functionality was provided by the manufacturers.

A simplified pseudo-code for the Arduino implementation is shown below:

```
setup() {  
    initSensors()  
    Bluetooth.startAdvertising()  
}  
loop() {  
    if (uart.available()):  
        command = Serial.readBytes()  
    if (command == transmitSensorData):  
        readSensorData()  
        formatSensorData()  
        sendSensorData(bytes)  
}
```

On startup, the microcontroller is advertising (making itself visible to other devices). When connected, it waits for a message to start or stop the transmission of sensor data. Then, if transmitSensorData is set to true, it starts transmitting the sensor data via the UART service.

The default capacitive touch sensor configuration proved to be not sensitive enough to detect touch over a non-conductive layer reliably.

As such, the thresholds for touch detection were reduced with a provided library function. The default thresholds (12 for touch, 6 for release) were changed to setThresholds(4,4).

While this helped, the touch detection was still unreliable. As such, the sensor configuration had to be changed. The Configuration register(ref) was changed in the Arduino library, from the default 16  $\mu$ A electrode charging current to 63  $\mu$ A (max allowed).

```
writeRegister(MPR121_CONFIG1, 0x3F); // default 0x10, 16uA, 6 // current: 63uA, 6
```

Figure 4.13 Sensor configuration code

The increase in charging current leads to a higher voltage on the electrode after the charging process, thus increasing the sensitivity(ref).

While increasing the number of samples taken for each level of filtering and electrode charging time increased the accuracy of the sensors, they made detection slower, and as such these values were not changed.

After some trial runs, it was found that the changes mentioned above were still not enough for accurate touch detection. It was found that the plastic over the conductive areas causes the baseline raw (analog) sensor readings, which usually change slowly depending on the changes in the environment, to not adjust quickly enough when a sensor is touched, which causes the following touches to not be detected properly. To fix this, the Filter Delay Limit register was changed from 0 to 255 (0xFF):

```
writeRegister(MPR121_FDLF, 0xFF); // original 0x00, changed to 0xFF
```

Figure 4.14 Sensor FDFL configuration code



After this change, the baseline became very stable and the touch sensitivity was increased dramatically. The sensors became so sensitive, that the touch thresholds had to be set back to their default setting.

Because BLE (Bluetooth low energy) does not have serial over Bluetooth functionality, the manufacturer provides a Nordic UART Service (NUS), that emulates a serial port over BLE. This proprietary service was used for Bluetooth communication.

Information from each sensor is stored in arrays of 5 bytes. This includes the state of the sensor (Touched/ not touched), the raw baseline readings, and filtered raw sensor readings:

State	Baseline readings		Filtered readings	
1	2	3	4	5

These arrays are then added together and sent as messages to the client. In this case, the number of sensors is 12, so the messages were 60 bytes long.

Although knowing the state of the sensors is enough for most applications, raw readings can provide information on the actual change in capacitance. This could identify gestures such as hover, as hovering a hand above the sensor will create a smaller change in capacitance than touching. The raw data can also provide insight into whether the user is touching the sensor with a single finger or if they are touching it with their palms.

#### 4.7.2 Gesture detection software (Creating and training the model)

Gesture detection was achieved by using a recurrent neural network written in python with TensorFlow<sup>2</sup>.

First, the data is read from the CSV files. As the data readings from disabled sensors are represented by -1, they had to be converted. I experimented by adding the data from disabled sensors as noise, by converting the readings to a random distribution of 1s and 0s, but converting these values to all zeros seemed to work best.

Then, sensors that have been touched for over 40% of recording time are zeroed:

```
inputs_train = inputs_train.reshape((inputs_train.shape[0], num_sensors, samples_per_gesture), order = 'F')

for a in range(shape_train[0]):
    for x in range(num_sensors):
        if(np.average(inputs_train[a][x]) >= 0.4):
            inputs_train[a][x] = np.zeros(samples_per_gesture)

inputs_train = np.transpose(inputs_train, [0,2,1])
```

Figure 4.15 Code for zeroing noise

This is done to remove unnecessary noise which occurs from a sensor being activated throughout the recording, without being involved in the touch sequence. For example, when the user is holding the cube in their hands while performing the gesture, or when the cube is placed on a surface that causes the bottom sensors to activate.

The model itself consists of 5 layers:

<sup>2</sup> <https://www.tensorflow.org/>

```

model = tf.keras.Sequential()
# Feature extraction
# Make the sequences artificially longer and reduce timesteps by max + min pooling
# Remove empty timesteps with masking
model.add(tf.keras.layers.MaxPool1D(pool_size = 3, strides = 1))
model.add(tf.keras.layers.Lambda(lambda x: -tf.nn.max_pool1d(-x, ksize = 5, strides = 3, padding = 'VALID')))
#model.add(tf.keras.layers.Masking(mask_value=0.)) # Could not be used with TensorflowSharp

# Add some noise to prevent overfitting
model.add(tf.keras.layers.GaussianNoise(0.1))
# Gated recurrent unit layer
model.add(tf.keras.layers.Bidirectional(tf.keras.layers.GRU(num_gestures * 3)))
# Output layer
model.add(tf.keras.layers.Dense(num_gestures, activation = 'sigmoid'))

```

Figure 4.16 Neural network model code

First, the data goes through feature extraction. These consist of max and min pooling layers. These are typically used to down-sample and extract features from images by partitioning them into a set of non-overlapping rectangles and, for each such sub-region, outputs the maximum/average or minimum.

In this case, max-pooling is used to increase the sequence length by expanding the number of ones in the sequences. This also smooths out the sequences, where any stutters are removed. As max-pooling decreases the number of lows (Timesteps where the sensors are not touched) between touches, min pooling had to be performed to increase these gaps between touches. Setting the strides to 2 also down-sampled the data from 200 timesteps to 96. Pool sizes were selected by experimentation.

As sequences usually occur during a small window of time in the recording, most timesteps are useless and do not contain any information on the sequence. As such, a masking layer can be used to skip empty timesteps. This further decreases the number of timesteps to be processed and increases the accuracy and time efficiency of the model. However, this layer created problems with the exported model graph in unity due to compatibility issues, and as such could not be used.

The main processing is done via the Gated recurrent unit (GRU) layer. I chose GRU, as it performed faster than an LSTM while maintaining a similar accuracy (Junyoung Chung, 2014). The best accuracy was achieved with a unit count of 3 \* no. labels. A small dropout of 0.03 was added to this layer to prevent overfitting but would have to be removed due to compatibility issues with TensorflowSharp. Instead, a Gaussian noise layer was added.

RMSprop optimizer was used, with a learning rate of 0.02. Nadam and Adam optimizers were also tested, but they were slightly worse in terms of prediction accuracy. Learning rates up to 0.03 can be used to improve the speed at the cost of less stable changes between epochs. A learning rate schedule was used to decrease the learning rate over time, as shown below:

```

epochs = 50
decay = 0.08 / epochs
def learning_rate_decay(epoch, learning_rate):
    return learning_rate * 1 / (1 + decay * epoch)

```

Figure 4.17 Code for decaying learning rate

Categorical cross-entropy was used as the loss function and Softmax was used as the activation for the output layer.

The model is trained over 50 epochs with a batch size of 32. The batch size can be increased to increase speed, but it will reduce the maximum accuracy that could be achieved and the model will converge slower, requiring more epochs. The epoch count can also be decreased, at the cost of stability.

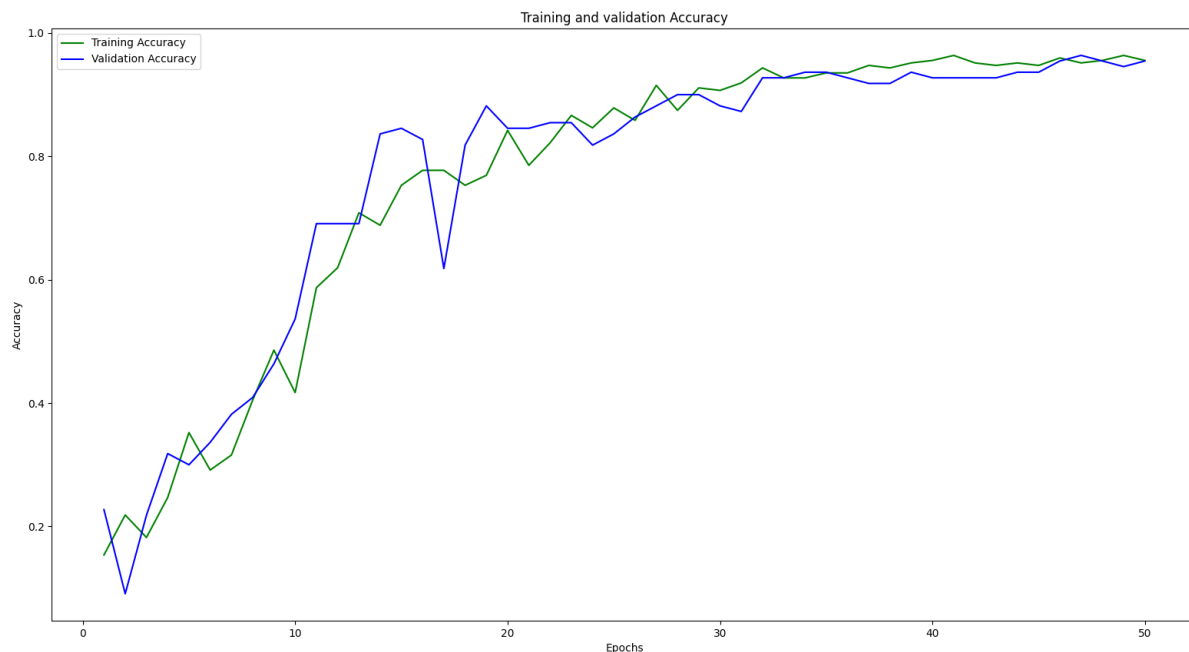


Figure 4.18 Training and Validation accuracy graph

As can be seen from the graph above, the categorical accuracy of the model increases at a very fast rate in the beginning and is not stable, as is usual with higher learning rates. Then, as the learning rate decreases, the model converges and to a stable accuracy of >90%. Average accuracy of 93% while testing the model was recorded. The validation accuracy closely follows the training accuracy, so it can be concluded that the model generalizes well.

When training is complete, the model is exported and saved as a frozen graph. The python program was packed into a standalone executable via pyinstaller<sup>3</sup>

#### 4.7.3 Client software – Bluetooth / Gesture Detection

Neither serial over Bluetooth nor Bluetooth Low Energy (BLE) is supported directly by unity. As such, Windows Runtime (WinRT) needs to be used to communicate with these devices. However, you cannot call WinRT APIs directly from C#. To be able to use BLE in unity, the BLE APIs need to be packed into a Dynamic-link library. I used a DLL from a public repository in this project<sup>4</sup>. It should be noted that this will only work on windows.

All Bluetooth Low Energy devices use the Generic Attribute Profile (GATT). It defines the way that two Bluetooth Low Energy devices transfer data back and forth using concepts called Services and Characteristics (Townsend, 2014).

Once BLE was working in unity, serial communication via UART was achieved by subscribing to the UART and using its read/write characteristics to send and receive data:

```
private static readonly string UARTServiceId = "{6e400001-b5a3-f393-e0a9-e50e24dcca9e}";
private static readonly string UARTReadCharacteristic = "{6e400003-b5a3-f393-e0a9-e50e24dcca9e}";
private static readonly string UARTWriteCharacteristic = "{6e400002-b5a3-f393-e0a9-e50e24dcca9e}";
```

Figure 4.19 UART IDs

3 different commands can be sent to the microcontroller. The client is able to reset the cube (Restarts the sensors), stop, and start the transmission of sensor data (To potentially save battery power).

<sup>3</sup> <https://www.pyinstaller.org/>

<sup>4</sup> <https://github.com/adabru/BleWinrtDll>

The data is stored in unity in structs for easy access. The structs contain the states of the sensors, and the change in raw sensor readings, which is calculating by subtracting the baseline readings from filtered readings.

Data recordings are stored in .csv files. As I did not use the raw sensor readings for gesture detection, only sensor states were stored. Each recording contains 200 rows (~3s) of sensor data.

```
t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11,t12
1,1,1,1,0,0,0,0,1,1,1,0
0,1,1,1,1,1,0,0,1,1,0,0
```

Figure 4.20 Recordings' format

The trained gesture classification model is loaded as a frozen graph. During detection, the data is first filtered by replacing columns where the state is 'touched' for more than 40% of the recording with zeroes. An assumption is made that, if the sensor is active for more than 40% of the recording time, it means that the cube is being held by touching that sensor and that the sensor is not part of any touch gesture.

After filtering, the data is sent to the model by using TensorFlowSharp<sup>5</sup>. TensorFlowSharp is a runtime that allows for tensorflow models to run from C#. Overlapping windows of data to enable continuous detection. Each window of data consists of 100 rows of the previous window + 100 rows of new data.

TensorflowSharp has been last updated in 2019 and currently does not support TensorFlow 2. Currently, Unity is developing its own neural network inference library, Barracuda<sup>6</sup>. However, it is still being developed and as such has very limited recurrent neural network support.

#### 4.7.4 Client software – UI

One of the most important features of the UI is the virtual representation of the printed model. The virtual model is hollow and is only comprised of the touch-sensitive areas of the object. Each area can be assigned to a sensor, which allows for the change of sensor state to be represented with visual graphics.

This also serves as a proof of concept of one of the possible uses of the interactive cube, as it could be integrated into video games or other software as a virtual object that can be manipulated from the physical world.

Additionally, the software allows for any model to be imported and used in the same way as the cube.

The UI of the client software can be broken into 3 windows:

- Bluetooth Connection
- Sensor setup
- Gesture detection/training

Firstly, the user scans for Bluetooth devices (see Figure 4.21). When the user clicks on

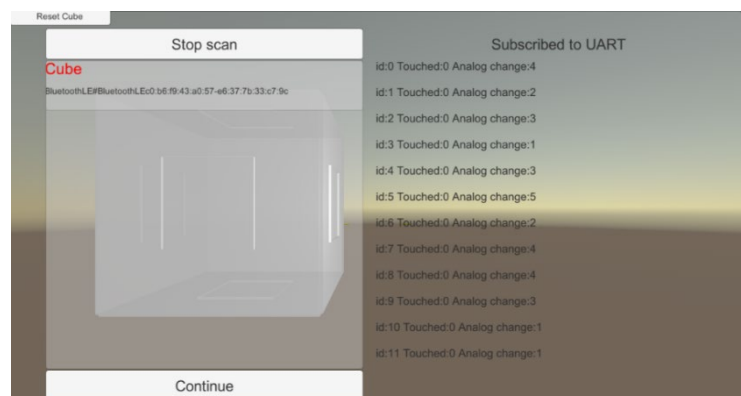


Figure 4.21 Bluetooth menu

<sup>5</sup> <https://github.com/migueldeicaza/TensorFlowSharp>

<sup>6</sup> <https://docs.unity3d.com/Packages/com.unity.barracuda@1.3/manual/index.html>

the device, the client automatically connects to the device and starts receiving sensor data from the UART service.

When the Bluetooth connection is established, the user has to go through the sensor setup (Figure 4.22, Figure 4.33). During sensor setup, the user maps the sensors to the virtual model of the cube.

This step is only required to accurately visualize touch via the virtual model. If the user has already gone through this step, they can simply load the previously set configuration. The touch plates on the virtual model change colour depending on the assigned sensor, making the process easier.

After attaching the sensors to the virtual model, the user is taken to the gesture control window (see Figure 4.24). Now, the user can add and record a new gesture. A keyboard key can be assigned to each gesture, which will be activated when the gesture is detected. Then, after retraining the model, the user can start the gesture detection again, which will now include the newly added gesture.

In this menu, the side panels of the virtual cube ‘light up’ in accordance to the touch sensors activated in the physical model.

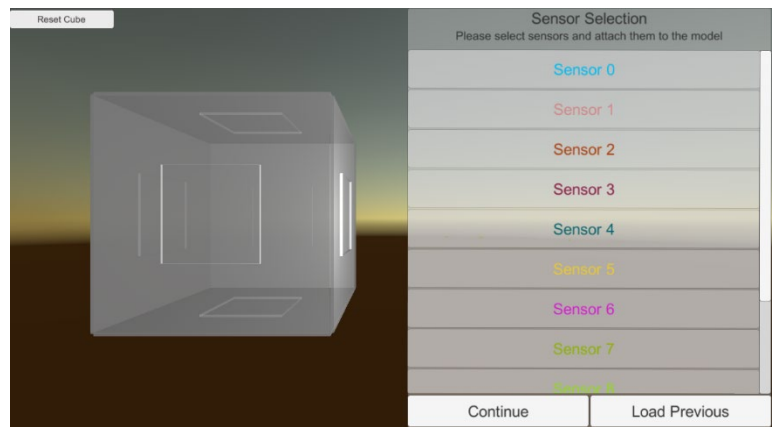


Figure 4.22 Sensor selection menu - no sensors selected

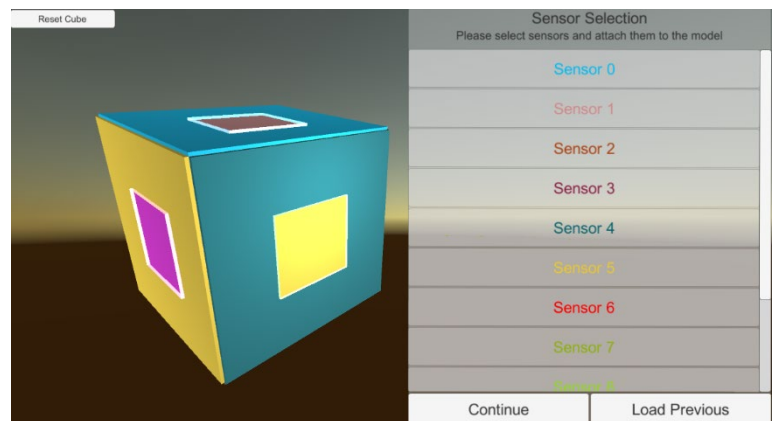


Figure 4.23 Sensor selection menu - all sensors selected

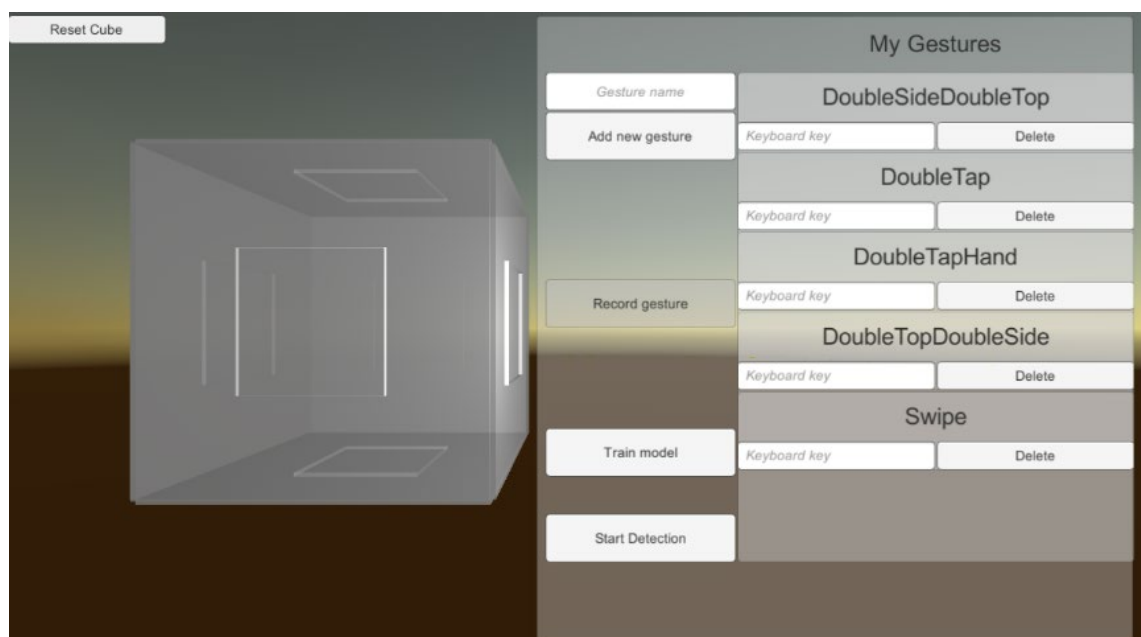


Figure 4.24 Gesture control menu

## 5. Testing and Evaluation

### 5.1 Sensor Testing

Individual testing has been performed to check the reliability of touch detection.

First, direct touch detection was tested by touching each of the conductive plates directly 10 times.

As the conductive plates are not exposed in the actual device, the same test was performed again, this time over a 1mm gap of non-conductive plastic.

In both cases, the sensors were able to detect touch with 100% accuracy.

### 5.2 Gesture Detection Study

To objectively measure the accuracy of touch gesture detection, an observational study was performed. Due to COVID-19 related restrictions, 4 participants were involved in this study. The study did not target any specific age group or demographics.

5 pre-trained touch gestures were tested. The pre-trained gestures are as follows:

- Simple gestures, where only one side panel is used:
  - Double Tap Mid – Touch the middle section of the top panel two times in quick succession
  - Double Tap Whole hand – Touch the whole top panel two times in quick succession
  - Swipe – Swipe through the top panel using up to 3 fingers.
- Complex gesture, which consists of multiple Simple gestures or is performed using multiple sides of the cube:
  - Double Top -> Double Side – Touch the top panel two times in quick succession, then touch one of the side panels two times in quick succession.
  - Double Side -> Double Top - Touch one of the side panels two times in quick succession, then touch the top panel two times in quick succession.

The testing procedure was as follows:

1. The participant is informed of the touch gesture they will be performing, an example is shown by the study conductor.
2. The participant then presses the space button on a computer keyboard to start recording the gesture.
3. The gestures menu is then highlighted in green, indicating that the actions are being recorded.
4. The participant performs the gesture during the recording time window, which is 200 recordings or ~3 seconds.
5. The predictions are recorded as gesture probabilities in a .csv file.

This procedure is repeated 10 times for each gesture with only the required sensors enabled (noiseless), and 10 times with all sensors enabled (with additional noise).

After testing the pre-recorded touch gestures, each participant was asked to record a custom gesture of their own choice (A total of 40 recordings for training). The procedure for training follows steps 1-4 of the testing procedure, but instead of predictions, raw input data is saved to .csv files. Then, the study conductor trains the neural network on the data gathered from the



recordings. Finally, the participant was asked to test the accuracy of the custom gesture by following the testing procedure of the pre-trained gestures.

A custom gesture is any gesture created by a user that expands upon a pre-recorded gesture (ex. Triple tap, which expands on Double tap) or is a combination of several gestures (ex. Swipe -> Double tap). The custom gestures were tested as a way to measure the ability of users to create and record gestures that are not pre-configured or tuned.

It is important to mention, that detection was not continuous during testing. The participants were asked to start the detection manually by pressing the space key on a keyboard and then complete the gesture within the 200 samples (~3s) window. As such, the accuracy of the gesture detection may be different in a real-world scenario, where the detection is continuous.

### 5.2.1 Results

Looking at the average gesture accuracies, the average accuracy of each gesture was above 70%. There was no major difference between simple gestures and complex gestures. A significant difference between recordings with and without noise was also not observed, as simple noise filtering was implemented into the system.

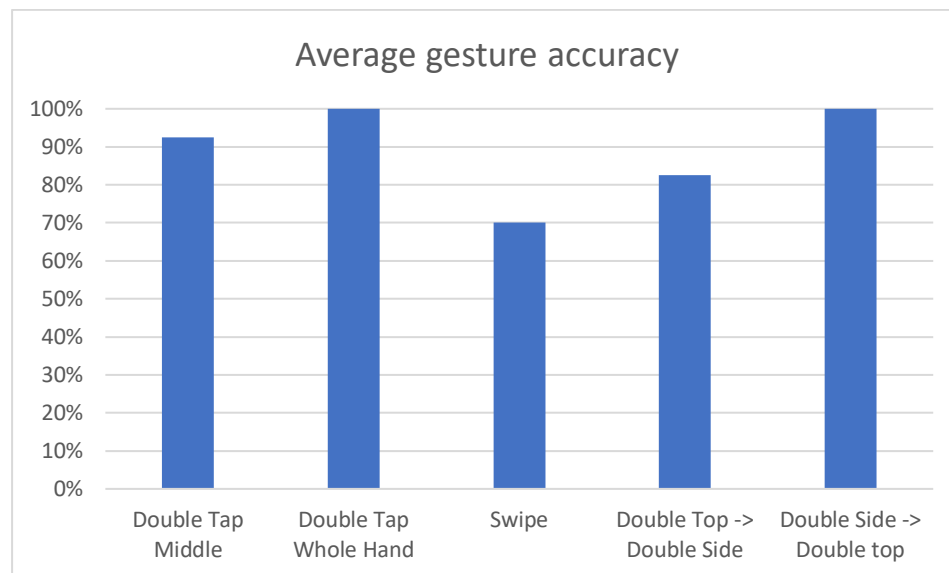


Figure 5.1 Average gesture accuracy

The accuracy of the swipe gesture was the lowest, with an average accuracy of 70% across all participants. During the study, it was noted that the average accuracy of gestures for one participant was significantly lower than for other participants. In particular, the average Swipe accuracy was only 20%, compared to the 80-100% score of other participants.

When looking at the graph of swipe gesture detection probabilities of this participant (fig 1), it is clear that most of the gestures (8/10) were classified as 'Other'.

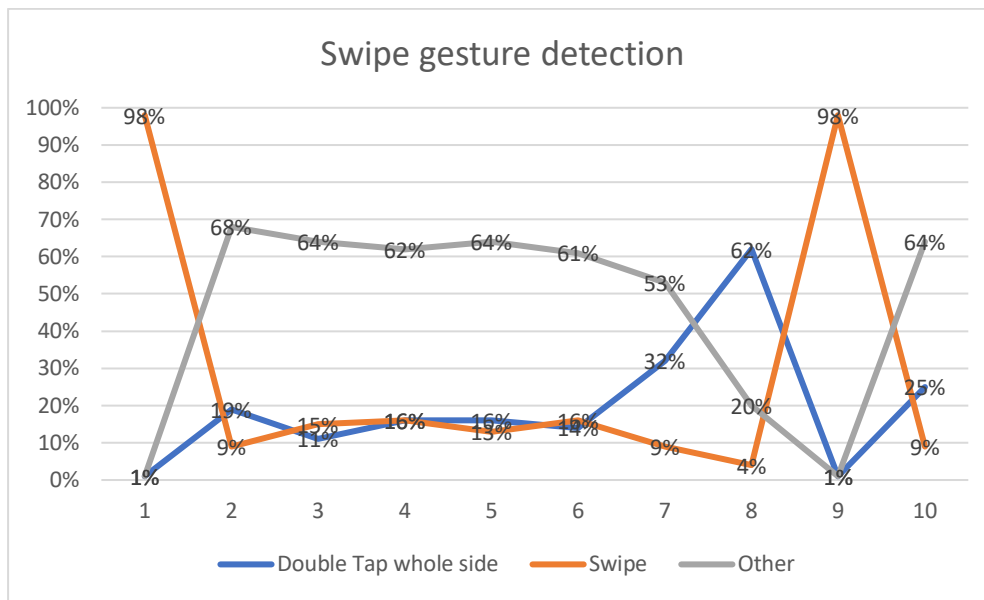


Figure 5.2 Accuracy of the 'Swipe' gesture

The 'Other' gesture contained data from various gestures that were not used as testing gestures. It was mainly used as noise. Most notably, it contained the single tap gesture, which was added as a measure to stop classifying single taps as double taps. After looking at the raw recordings and performing further individual testing, it was found that, under certain conditions, touching the middle sensor of a side panel can also activate the outer plate and vice versa. As such, when performing the swipe gesture under these conditions, all sensors on the panel were active and the touch gesture was, in practice, a tap.

This may also explain why the accuracy of other gestures was also lower for this participant, as due to the differences in human biology and human skin conditions (i.e skin is more conductive when it is wet), the sensors were more sensitive to touch.

This issue can be fixed by adjusting the sensitivity thresholds. Lowering the sensitivity was noted to have fixed the issue according to later individual testing.

The average accuracy of the custom gestures was 85%. No significant difference was observed between training with 20 and 40 recordings of the gesture, but this may be due to the lack of testing data or the number of similar gestures, as all custom gestures were fairly distinct.

### 5.3 Static Program Analysis

For the python code (Machine Learning code), pycodestyle<sup>7</sup> was used to check whether the code follows the PEP 8 python style guidelines.

Multiple code style issues were found, which were subsequently fixed.

For the Unity C# code, visual studio code analyser was used to check the code for styling errors.

```
train.py:208:80: E501 line too long (95 > 79 characters)
train.py:209:1: E265 block comment should start with '#'
train.py:227:1: E302 expected 2 blank lines, found 0
train.py:235:1: E303 too many blank lines (6)
train.py:237:80: E501 line too long (82 > 79 characters)
train.py:240:80: E501 line too long (103 > 79 characters)
train.py:241:1: E265 block comment should start with '#'
train.py:241:80: E501 line too long (91 > 79 characters)
train.py:251:56: E251 unexpected spaces around keyword / parameter equals
train.py:251:58: E251 unexpected spaces around keyword / parameter equals
train.py:251:77: E251 unexpected spaces around keyword / parameter equals
train.py:251:79: E251 unexpected spaces around keyword / parameter equals
train.py:251:80: E501 line too long (231 > 79 characters)
train.py:251:107: E251 unexpected spaces around keyword / parameter equals
train.py:251:109: E251 unexpected spaces around keyword / parameter equals
train.py:264:1: E303 too many blank lines (3)
train.py:271:1: E302 expected 2 blank lines, found 1
train.py:271:80: E501 line too long (88 > 79 characters)
train.py:283:80: E501 line too long (92 > 79 characters)
```

Figure 5.3 Initial pycodestyle warnings

<sup>7</sup> <https://github.com/PyCQA/pycodestyle>



## 5.4 Battery Testing

Battery testing was performed by using the following procedure:

1. The battery is fully charged. The 110mAh battery charge time at 200mA is around 33 minutes, but it is charged for 4 h to ensure that it is fully charged.
2. Then, the cube was set to transmit data to the client continuously.
3. When data transmission stops due to the device running out of battery power, the elapsed time is measured.

The test was stopped after ~8h of continuous data transmission, as 8h of battery life was deemed sufficient according to the requirements.

## 5.5 Reusability Testing

The reusability of the software was tested by using a different 3D model with different numbers of touch sensors in the client application.

A 3D model of the first prototype was used, which contained 5 different touch areas. The application functioned correctly.

It was not possible to reliably test the application with a model that uses more than 12 touch sensors due to hardware limitations. However, setting the sensor amount to 16 did not crash the application or cause any unexpected behaviour to occur.

## 5.6 UI Testing

Testing the User Interface was performed by following this procedure:

1. All UI buttons are pressed and checked for unexpected behaviour
2. A large number of elements (20) are added to the UI lists and available actions are performed and checked for unexpected behaviour.

To ensure reliability, the tests were performed 3 times.

## 5.7 Fulfilment of Requirements

Most requirements have been fulfilled. Due to the lack of gyroscope and accelerometer modules in the final prototype, the device is not able to measure acceleration or rotation.

The touch sensitivity thresholds may need to be adjusted depending on the person who is using the device, as discussed in the gesture detection study results. As such, it cannot be stated for sure whether the device will detect touch input reliably in all human finger capacitance conditions.

### 5.7.1 Functional requirements

ID	Description
1	The touch interface shall be able to detect touch input
1.1	The touch interface shall be able to detect touch input over a 1mm distance
2	The cube shall have at least one sensor on each side
3	The cube shall be able to sense acceleration
4	The cube shall be able to sense its rotation

5	The cube shall be able to transmit sensor data via Bluetooth
6	Touch gestures shall be detected
6.1	Simple gestures shall be detected
6.2	Complex gestures shall be detected
7.	The user shall be able to set custom touch gestures
8.	The GUI shall be able to display the actions performed on the interface on a virtual 3D model.

### 5.7.2 Non-Functional requirements

ID	Description
1.	The touch interface shall be able to detect touch input in all human finger capacitance conditions
2.	Touch gestures shall be detected with 70% accuracy
2.1.	Touch gestures shall be detected with 80% accuracy when there is no noise from other sensors (When the interface is static)
2.2.	Touch gestures shall be detected with 60% accuracy when there is noise from other sensors (When the object is being held in hands)
3.	The battery should last at least 7 hours when the interface is in use.
4.	The Software shall support the addition of more than 12 sensors.
5.	The Software shall be able to support custom 3D models.
6.	The user shall be able to set custom touch gestures

## 6. Conclusion

This final chapter reviews the aims set forth at the start of the project, discusses the challenges faced throughout the project, and provides suggestions on future work in this area of research.

### 6.1 Review of aims

#### 6.1.1 Designing and printing an interactive interface

A 3D model was successfully designed and printed. The model has been designed by following the requirements set forth at the beginning of the project. However, the model currently does not allow for the addition of other modules (Gyroscope/ Accelerometer) without minor changes.

#### 6.1.2 Writing software for the interpretation of touch gestures

Arduino code was written, which converts the data gathered from the touch sensors to a format suitable for transmission over Bluetooth. The sensors have been properly configured to work reliably with the printed touch panels. However, the interpretation of the sensor data was moved from the microcontroller to the desktop application.

#### 6.1.3 Exploring the use cases of the printed object

A demo client software was designed, which showcases one of the use cases of the touch interface. It allows for a user to record custom touch gestures and use them to control various applications via keyboard events. Even though demos fully showcasing other use cases were not developed, the developed demo includes a virtual representation of the printed model that could potentially be controlled in software by using the physical model as a control, which gives insight into the VR/AR uses for such a device.

#### 6.1.4 Simplifying the process of creating 3D printed touch interfaces/gesture detection software

Although the success of this aim can only be measured subjectively, I believe that it was fully achieved. The project provides written guidelines on designing printed capacitive touch plates and printing with conductive filament in general. An optimal sensor configuration was also discussed, which may help when calibrating the touch sensors to work with a touch plate configuration that has the conductive areas hidden beneath a non-conductive layer of plastic.

A potential gesture classification model was provided, which can be used in projects involving similar sequence classification.

The client program is also fully customizable, allowing for different 3D models to be used, involving a different number and configuration of capacitive touch sensors.

## 6.2 Technical and Personal Challenges

As mentioned in the design chapter, I initially planned on using dual extrusion technology for the project. However, due to the COVID-19 related restrictions, I was unable to use the university facilities and as such had to use my DIY single extrusion 3D printer. This proved to be a great technical challenge, as it meant that conductive and non-conductive parts of the model had to be printed separately and then combined via other means, which increased the complexity of the project. As my 3D printer was of low quality and has not been used for several years, it

was an interesting challenge of restoring and upgrading it to make it suitable for printing larger scale models.

Another difficulty that arose from the COVID-19 pandemic was performing a gesture detection study. As the study consists of testing hardware, it needs to be in-person. However, this was impossible due to the various restrictions. As such, the study was very limited in scope and only involved participants that live in the same household.

One of the personal challenges I faced during this project is the lack of experience in 3D modelling. The only 3D modelling experience I had until this project was adding holes to already existing models. As such, I had to start from scratch and faced some issues, which can be reflected in the complete lack of hierarchy in my 3D modelling environment.

Software incompatibility was also a huge issue. Most of the processing was done in the same program that provides the UI, Unity, to decrease the complexity of the project. However, this created issues, as Unity does not have native any Bluetooth Low Energy support as of now. Additionally, the tools provided by unity for running neural networks are very limited. An outdated TensorFlowSharp runtime had to be used, which required an outdated version of TensorFlow to be used, which required an outdated Python version to be used, which in turn requires for the whole gesture detection model software to be run in a virtual environment. However, packing the Python software into an executable seemed to have alleviated the issues.

### 6.3 Future Work

There are several areas in this project that could be further worked upon.

Firstly, as the 3D design in this paper focused on single extrusion 3D printing, it would be interesting to explore the improvements that could be made to the model by multi-material printing. With my design, the model is not fully 3D printed, as it still requires parts to be printed and attached separately. However, by using this technology, it could be possible to print the entire object in one session, where only the electronic parts would need to be attached. Replacing traditional wires that were used in this device with printed conductive plastic wires could also be investigated. This would further reduce the complexity of the model assembly process.

Increasing the number of touch sensors per panel, as mentioned in the sensor configuration table in the design section, could potentially allow for more complex gestures to be performed on a single panel. The design that was chosen was only able to support Swipe and tap gestures but having 4 or more sensors per side could allow for gestures such as scale and rotation. It might also be interesting to pursue the printing of capacitive sensor grids instead of separate touch plates, similar to those found in touchscreens.

Additionally, it would be interesting to see other sensors included in the device. The addition of other sensors could open more interaction opportunities. For example, a gyroscope and an accelerometer could allow for full 3D manipulation of the object. This would open the device for more use cases, especially in Augmented Reality applications, as mentioned in the design section of the report.

Finally, the implementation of continuous gesture detection in this project was quite poor. It is not reflected in the study results, as the studies were performed on well-segmented recordings of sequences. However, in the real world, the sequences will never be segmented perfectly, as the computer is receiving a stream of data, where the sequences can occur at any time. As such, the handling of the data stream could be greatly improved upon and it would be interesting to see how that could be achieved.

## 6.4 Closing Remarks

Overall, the project has been successful. I was able to design and produce a fully functional device along with creating working software for said device.

This has been one of my most challenging projects. Not only was the scope of the project large, it involved experimenting with methods and techniques that have not yet been thoroughly explored.

As it required working with different fields of computer science and engineering, it provided me with a unique learning experience. I greatly improved my knowledge in multiple fields, such as 3D design, 3D printing, machine learning, and Unity development.

In the end, I would say that I greatly enjoyed working on this project and I believe that the experience I gained will prove useful with similar future projects.

## 7. References

- All About Circuits, 2016. *Introduction to Capacitive Touch Sensing*. [Online]  
Available at: <https://www.allaboutcircuits.com/technical-articles/introduction-to-capacitive-touch-sensing/>  
[Accessed 2020].
- All3DP, 2021. *Conductive Filament (PLA): The Basics & Best Brands*. [Online]  
Available at: <https://all3dp.com/2/conductive-filament-brands-compared/>  
[Accessed 2021].
- Autodesk, 2018. *Additive vs. subtractive manufacturing – what’s the difference?*. [Online]  
Available at: <https://blogs.autodesk.com/advanced-manufacturing/2018/07/29/additive-vs-subtractive-manufacturing-whats-the-difference/>  
[Accessed 2020].
- Berman, B., 2012. 3-D printing: The new industrial revolution. *Business Horizons*, 55(2), pp. 155-162.
- C. Shemelya, F. C. E. A. E. M. J. R. D. E. D. M. R. W. E. M., 2013. *3D printed capacitive sensors*. Baltimore, MD, USA, s.n.
- Digi-Key, 2018. *How is a PCB Manufactured*. [Online]  
Available at: <https://www.digikey.com/en/maker/blogs/2018/how-is-a-pcb-manufactured>  
[Accessed 2020].
- Du, L., 2016. *An Overview of Mobile Capacitive Touch Technologies Trends*, s.l.: Cornell University.
- Fishkin, K. P., 2004. A taxonomy for and analysis of tangible interfaces. *Personal and Ubiquitous Computing*, 8(5), pp. 347-358.
- Fitzmaurice, G. W. a. I. H. a. B. W. A. S., 1995. *Bricks: Laying the Foundations for Graspable User Interfaces*. ACM Press/Addison-Wesley Publishing Co., Denver, Colorado, USA.
- Grand View Research, Inc, 2020. *3D Printing Market Size, Share & Trends Analysis Report By Material, By Component (Hardware, Services), By Printer Type (Desktop, Industrial), By Technology, By Software, By Application, By Vertical, And Segment Forecasts, 2020 - 2027*, s.l.: s.n.
- Grosse-Puppendahl, T. a. H. C. a. C. G. a. W. R. a. B. O. a. H. S. a. R. M. S. a. S. J. R., 2017. *Finding Common Ground: A Survey of Capacitive Sensing in Human-Computer Interaction*. New York, NY, USA, Association for Computing Machinery.
- Hideyuki Suzuki, H. K., 1995. *AlgoBlock: a Tangible Programming Language - a Tool for Collaborative Learning*. Bloomington, IN, USA, Association for the Advancement of Computing in Education.
- Hiroshi Ishii, B. U., 1997. *Tangible Bits: Towards Seamless Interfaces*. Cambridge, MA 02139-4307 USA, Conference on Human Factors in Computing Systems.
- Ishii, H., 2008. The tangible user interface and its evolution. *Communications of the ACM*, 51(6), pp. 32-36.
- Jean-Baptiste de la Rivière, C. K. E. O. N. D., 2008. *CubTile: a multi-touch cubic interface*. Bordeaux France, s.n.
- Junyoung Chung, C. G. K. C. Y. B., 2014. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*, Montréal: NIPS 2014 Deep Learning and Representation Learning Workshop.
- Kaufui V. Wong, A. H. F. F., 2012. A Review of Additive Manufacturing. *ISRN Mechanical Engineering*, Volume 2012, p. 10.

- Kevin Lefevre, S. T. M. S. A. K. A. B. A. B., 2018. *Bricks, Blocks, Boxes, Cubes, and Dice: On the Role of Cubic Shapes for the Design of Tangible Interactive Devices*. Hong Kong, China, s.n.
- Mackey, B., 2011. 43.1: Invited Paper: Trends and Materials in Touch Sensing. *SID Symposium Digest of Technical Papers*, 42(1), pp. 617-620.
- Martin Schmitz, M. K. M. B. R. L. M. M. J. S., 2015. *Capricate: A Fabrication Pipeline to Design and 3D Print Capacitive Touch Sensors for Interactive Objects*. Charlotte NC USA, s.n.
- McNerney, T. S., 2000. *Tangible programming bricks : an approach to making programming accessible to everyone*, Cambridge, Massachusetts, USA: MIT.
- Piper, B. a. R. C. a. I. H., 2002. *Illuminating Clay: A 3-D Tangible Interface for Landscape Analysis*. New York, NY, USA, Association for Computing Machinery.
- Seeed, 2020. *Touch sensors: What it is, How it works, Arduino Guide*. [Online]  
Available at: <https://www.seeedstudio.com/blog/2019/12/31/what-is-touch-sensor-and-how-to-use-it-with-arduino/>  
[Accessed 2020].
- Technology Outlet, 2019. *Best Budget 3D Printers (Under £400)*. [Online]  
Available at: <https://technologyoutlet.co.uk/blogs/news/best-budget-3d-printers-under-400>  
[Accessed 2020].
- Townsend, K., 2014. *Introduction to Bluetooth Low Energy*. [Online]  
Available at: <https://learn.adafruit.com/introduction-to-bluetooth-low-energy>  
[Accessed 2020].

## 8. Appendix

### 8.1 Project proposal

# Conductive 3D printing for touch interfaces

Martynas Dabravalskis

---

## Abstract

The proposed project reported in this paper will aim to explore the usage of conductive 3D printer filament in combination with regular filament to create interactive 3D printed objects with integrated capacitive touch areas. Even though 3D printing has become increasingly popular over the last few years, the objects created are still very simple static objects. As such, the usage of conductive filament should greatly increase the interactivity of 3d printed products.

The first step of the project will involve designing and printing a simple geometric object with conductive touch areas. After said object is printed, a microcontroller will be used to detect touch input. The controller will be programmed to interpret various gestures. The product will be tested for several use cases, such as AR (Augmented Reality), and as a controller device for various types of software. The size and other limitations of such an object will also be analysed.

## 1. Introduction

3D printing has become increasingly popular over the last few years. As the technology continues to mature and 3D printer production and operating costs continue to decline it is estimated that the 3D printing market size will continue to grow at a compound annual growth rate (CAGR) of over 14% for the next seven years (Berman, 2012).

3D printing enables small quantities of customised goods to be produced at relatively low costs (Grand View Research, Inc, 2020). This allows for rapid production of customizable products without the need of expensive and time-consuming development of full production lines.

However, most products are limited by the materials used to create them. As 3D printing usually involves a single type of plastic or other material used to print the product, the products created are passive and do not support any type of interactivity by themselves.

Due to this limitation, the product's interaction would normally be introduced by external means, such as adding copper touch plates or other sensors to the object after it has been printed. However, the new availability of conductive printing materials as well as the availability of multi material printing technology should allow for the printing of interactive products. A typical plastic filament, such as an ABS or PLA plastic, can be used in combination with conductive 3D filament to print objects with integrated conductive touch plates, as well as internal wiring to enable easy access to the sensor plates.

The proposed project will aim to explore the possibilities of interactive objects printed using the method mentioned above. The project will involve creating a fully interactive 3D printed cube with integrated touch plates. The shape of such interactive object was chosen to be a cube, as cubic shapes allow for wide variety of interaction possibilities (Kevin Lefeuvre, 2018). A few of these interactive cubes of varying sizes will be printed to investigate the size limitations of such an object.



To explore the interactivity of the designed product, the object will be designed to allow for various touch gestures to be registered via the use of a microcontroller board with custom software. The paper will explore how different touch sequences or gestures can be used as control inputs for different applications.

## **2. Background**

Previous projects relating to the proposed project include a paper on designing and 3D printing of capacitive touch sensors for interactive objects (Martin Schmitz, 2015). The paper reports on how to incorporate 3D printed touch surfaces in static 3d printed objects. The paper reports on the implementation of such sensor surfaces, as well as their technical limitations.

However, even though this paper does present several examples and the applications of the 3d printed objects with integrated capacitive touch sensors, it does not explore the interaction possibilities in more detail. The paper does not expand on the implementation of complex touch gestures and how they can be used to interact with the object.

Another related paper focuses on the performance of different types of 3D printed touch sensors (C. Shemelya, 2013). The researchers in this paper incorporated copper wire into 3D printed objects to allow for touch input to be perceived. The results show that it is possible to encapsulate the conductive parts of the touch sensor within the 3D printed plastic material.

However, this paper did not explore the possibilities of using conductive plastic itself to create the capacitive surfaces. The expensive and time-consuming process of adding copper wires within the 3d printed object, a conductive plastic filament can be used to replace the additional copper wiring and print the completed object with the conductive plates in one printing session.

The decision of choosing a cubic shape as an interactive prototype was supported by a paper which highlights the advantages of cubic shapes for designing interactive devices (Kevin Lefevre, 2018). The paper discusses various properties of cubic shapes that make them a prevalent choice in designing interactive interfaces. The paper also mentions how the simple geometric properties of a cube can be used to create interactive interfaces.

Adding to the previously mentioned research, this paper explores the touch interaction techniques of cubic shaped interfaces in more detail. The touch surfaces on the cube allowed for the manipulation of virtual 3D objects by implementing common multi-touch gestures for translation, scale and rotation of the virtual object. The 3D multi-touch interface proposed in this paper could possibly be created with the use of printed conductive touch sensors (Jean-Baptiste de la Rivière, 2008).

## **3. The Proposed Project**

### **3.1 Aims and Objectives**

The aim of the project is to explore the usage of conductive 3D filament in creating a fully interactive touch interface as well as explore the interaction techniques that can be achieved by utilizing the integrated capacitive touch sensors.

In order to achieve these aims, the following objectives will be involved:

- Designing and printing the interactive interface – A custom 3D model will be created and printed for this project. Separate capacitive touch areas will need to be included in the model, as well as the necessary wiring that would allow for the capacitive surfaces to be connected to

an external microcontroller board through a single access point on the cube model. A few different sized models, as well as models with different sensor placement will need to be created to optimize the object.

- Writing a software for the interpretation of touch gestures – A software will need to be written that would allow for the microcontroller to interpret the touch inputs received from the sensors. The software will be made to understand the various touch gestures and translate them into direct commands to a computer.
- Exploring the interaction possibilities of the printed object – Touch gestures will be implemented and explored, including the possibility of multi-touch gestures. A piece of software will be chosen or designed to showcase the capabilities of the touch-enabled device.

## 3.2 Methodology

In order to investigate the effectiveness of printed conductive 3D surfaces, a fully interactive interface will need to be designed. Capacitive touch plates will have to be integrated on each side of the cube model to allow for full interactivity. Several versions of the model will be created, differing on the placement and the amount of touch sensors (up to 12) and size, to investigate the effect these changes have on the effect of interactivity that can be achieved. Due to the technical limitations of conductive plastic filament, the technical findings from previous research will be followed when designing the capacitive touch plates (Martin Schmitz, 2015).

In order to perceive and interpret touch input, we will use a microcontroller and a capacitive touch sensor breakout board. Software for the microcontroller will need to be developed to interpret the touch gestures that will be used to interact with the interface. Touch gestures will be implemented in guidance to the proposed interaction technique mentioned in one of the relevant research papers (Jean-Baptiste de la Rivière, 2008).

For the design of the 3D model, we will use the TinkerCad software, as the program is really simple to use and has the necessary tools to make a 3D models that are suitable for double extrusion. As the 3D model should not be complex (A cubic shape), more powerful 3D modeling software may not be required. As an Ultimaker 3D printer is likely to be used, we will use the Cura software to slice the model and prepare it for 3D printing.

An Arduino board was chosen as a microcontroller, as it is simple to program and has an eco-system of many expansion boards with easy to use libraries. The Arduino IDE will be used to design the software.

The software engineering approach used to create the required software will be an iterative methodology. As this is an explorative project, the software requirements are expected to change due to new findings or the changes in hardware. As such, some of the project's stages may overlap.

## 4. Programme of Work

The project will begin early October 2020, running until March 2021, and it will be broken up into the following stages:

- Research – This stage will consist of researching different materials, 3D modelling and printing techniques for the 3D model. It will also involve researching for various touch interaction techniques for the model as well as ideas for the possible use cases of the finished product. This will take 2-4 weeks.
- 3D model analysis and design – This involves designing and producing several versions of the 3D model. The model will be created by combining regular and conductive plastic

filaments to create a cube shaped interface with integrated capacitive touch plates. Dual extrusion printing will be used to print these models. This stage will take 4-6 weeks.

- Software development – This involves creating software for the microcontroller according to the specifications of the printed objects. The software will be designed to interpret touch input from the capacitive sensors and detect gestures. This stage will take 4-6 weeks.
- Use-Case analysis – This stage will consist of exploring the usage the interactive interface as a controller device for software applications. This stage will take 4-6 weeks.
- Data Analysis and Evaluation – This stage will consist of analyzing and evaluating the data collected throughout the project. This will include a comparison of the different prototypes according to their technical and other characteristics. This stage will take 4-6 weeks.

As mentioned before, due to the nature of the project, the first three stages of the project may overlap.

## 5. Resources Required

Access to a 3D printer capable of dual extrusion will be required. This project will require the use of an Arduino Board, a breakout capacitive sensor board, and regular plastic (PLA/ABS) and conductive plastic 3D filaments.

## 6. References

- Berman, B., 2012. 3-D printing: The new industrial revolution. *Business Horizons*, 55(2), pp. 155-162.
- C. Shemelya, F. C. E. A. E. M. J. R. D. E. D. M. R. W. E. M., 2013. *3D printed capacitive sensors*. Baltimore, MD, USA, s.n.
- Grand View Research, Inc, 2020. *3D Printing Market Size, Share & Trends Analysis Report By Material, By Component (Hardware, Services), By Printer Type (Desktop, Industrial), By Technology, By Software, By Application, By Vertical, And Segment Forecasts, 2020 - 2027*, s.l.: s.n.
- Jean-Baptiste de la Rivière, C. K. E. O. N. D., 2008. *CubTile: a multi-touch cubic interface*. Bordeaux France, s.n.
- Kevin Lefevre, S. T. M. S. A. K. A. B. A. B., 2018. *Bricks, Blocks, Boxes, Cubes, and Dice: On the Role of Cubic Shapes for the Design of Tangible Interactive Devices*. Hong Kong, China, s.n.
- Martin Schmitz, M. K. M. B. R. L. M. M. J. S., 2015. *Capricate: A Fabrication Pipeline to Design and 3D Print Capacitive Touch Sensors for Interactive Objects*. Charlotte NC USA, s.n.

## 8.2 Research information sheet

LANCASTER UNIVERSITY DEPARTMENT OF COMPUTING

### RESEARCH INFORMATION SHEET

**TITLE OF RESEARCH:**

Conductive 3D printing for touch interfaces

**PRINCIPAL INVESTIGATOR:**

Martynas Dabravalskis - 34850546

**INTRODUCTION**

You will be taking part in a research study concerning the viability 3D printing in designing and manufacturing touch interfaces. The aim of this study will be to evaluate the effectiveness of 3D printed touch plates in sensing touch input and gestures via the use of Capacitive sensing.

We invite you to participate in a study that will involve the collection of objective data on tasks completed. As participants, we invite you to test the accuracy of the touch detection of the prototype object by completing predefined tasks. You will be provided with instructions on how to perform the tasks.

It is important that you read and understand several principles that apply to all who take part in our studies:

- a) taking part in the study is entirely voluntary;
- b) personal benefit may not result from taking part in the study, but knowledge may be gained that will benefit others;
- c) any significant findings will be discussed with you if you desire;
- d) you may withdraw from the study at any time.

The nature of the study, the risks, inconveniences, discomforts, and other pertinent information about the study are discussed below. You are urged to discuss any questions you have about this study with the investigator before you sign this consent.

In accord with all of our research protocols, privacy will be fully protected and confidentiality maintained at all times. See <https://www.lancaster.ac.uk/research/participate-in-research/data-protection-for-research-participants/>.

**BACKGROUND & PURPOSE:**

This research study is concerned with testing the viability 3D printing in designing and manufacturing touch interfaces. The aim of this study will be to evaluate the effectiveness of 3D printed touch plates in sensing touch input and gestures via the use of Capacitive sensing.

### **STUDY PROCEDURE:**

You are being asked to participate in a study that will require your cooperation in one or more of the following:

- Performing certain individual tasks as requested by the investigators. These may include but are not limited by:
  - Touching a touch sensor 10 times, with a 2sec delay between each touch
  - Performing a double touch 10 times on a specified part of the touch interface (Touching the sensor 2 times in quick succession)
  - Performing other touch gestures.
  - Creating a custom touch gesture, recording it for 25 times, and then testing it 10 times.
- Providing us with any remarks you may have about the detection system or the touch interface.

When writing the results from our testing into a project report or any other form of documentation, steps are taken to ensure anonymity for all those involved in the study. No personal details will be recorded. Confidentiality will be maintained at all times. Any recordings that are made are the property of the researcher and will be kept in a secure environment and destroyed at the conclusion of the research.

### **RISKS OF PARTICIPATION IN THE STUDY:**

The risks of participating in this study are minimal.

It is the investigators' intention that your identity in these studies will remain confidential.

Your particular contribution to the study – what you disclose during the study – will be anonymized.

### **BENEFITS:**

There may be no personal benefit to you from participating in this project. However, some personal benefits of this research may include learning more about. 3D printing, conductive 3D printing, touch sensing, touch gesture detection.

We believe this work can make an important contribution to current debates on whether additive manufacturing can be a suitable replacement for professional manufacturing tools when creating fully working touch enabled devices.

**COSTS AND COMPENSATION:**

You will not be paid for participating in this study.

There is unlikely to be any cost - financial or other - to you for participation in the study.

**CONFIDENTIALITY:**

All information collected in this study belongs to the fieldworker and will be maintained in a confidential manner at Lancaster University. Nobody, other than the fieldwork researcher, will have access to the data. Any identifiable data (including recordings of participants' voices) on portable devices (eg audio recorders, etc) will be erased from it as quickly as possible and in the meantime the device will be stored securely. Any recordings will be destroyed at the end of the project. Although rare, it is possible that disclosure may be required by law. Otherwise, the information will not be disclosed to third parties without your permission. If the study is published, your name will be kept confidential.

**PEOPLE TO CONTACT:**

If you have further questions related to this research study, you may contact Dr Steven Houben, Dept of Computing and Communications, Lancaster University

Email: [s.houben@lancaster.ac.uk](mailto:s.houben@lancaster.ac.uk)

Address: Computing Department, Infolab21, Lancaster University, Lancaster LA1 4WA

You may also if you wish contact an independent person about this research – specifically, Adrian Friday, Head of School.

School of Computing and Communications

(<http://www.scc.lancs.ac.uk/>)

InfoLab 21 Building, South Drive,

Lancaster University, Lancaster LA1 4WA, England

email: [a.friday@lancaster.ac.uk](mailto:a.friday@lancaster.ac.uk)

tel: +44 1524 510326

## 8.3 Research Participant Consent Form

### Research Participant Consent Form

---

**PROJECT TITLES:** Conductive 3D printing for touch interfaces

**INVESTIGATORS:** Martynas Dabravalskis

**PARTICIPANT NAME:** \_\_\_\_\_

**TITLE:** \_\_\_\_\_

---

I agree to participate in the project named above, the particulars of which have been explained to me. I have read the Research Project Description a written copy of which has been given to me to keep.

I understand that any information I provide is confidential, and that, subject to the limitations of the law, no information that could lead to the identification of any individual will be directly disclosed in any reports on the project, or to any other party.

I agree to being: (tick as appropriate):

- ☐ interviewed;
- ☐ observed;
- ☐ photographed;
- ☐ enrolled on a Facebook group;
- ☐ part of a focus group

I agree to the following data being collected:

- ☐ field notes;
- ☐ audio-recordings;
- ☐ photos;

- ☐ video-recordings;
- ☐ Facebook entries;
- ☐ other (e.g. Web histories) \_\_\_\_\_

I also agree to the data above being used for later analysis by the researchers above only. To preserve anonymity, I understand that all written work referring to this data will use pseudonyms for me unless written permission is later obtained. I also understand that direct access to the identity of participants is restricted to named researchers above only.



I acknowledge that:

- a) I have been informed that I am free to withdraw from the project at any time without explanation or prejudice and to withdraw data previously supplied.
- b) Participation in this project is voluntary.

Signature: \_\_\_\_\_ Date: \_\_\_\_\_  
(Participant)