UNIVERSITÄT
BAYREUTH

Fakultät Mathematik, Physik, Informatik
Institut für Informatik
Lehrstuhl für Serious Games
Prof. Dr. Jörg Müller

Big Master Project

The Haptic Printer

Author : **Anuj Sharma**,
Akshaya Kumar Krishnappa Ramakrishna

Supervisor : **Viktorija Paneva**

September 27, 2021
Version: Final

Abstract

Over the years of evolution of human computer interaction, we have seen various kinds of interaction techniques. The recent trend of contactless haptic technology has been on the rise. In our project we focus to render custom shapes on the user palm, using different kinds of rendering techniques. The main objective of this project is to map various kinds of user input like CSV, SVG, drawing from canvas and inputs from leap motion. We consume these inputs and process them in order to render custom shapes. Also, this report analyse and compare the different techniques of rendering with various validation techniques and deduce an overall result from the study and implementation of the proposed application.

Contents

1 Introduction

The haptic technology enables human computer interaction in a new way compared to previous existent interaction techniques. Haptics gives us a new dimension of interaction technique without any physical contact between the user and the device. This contactless haptic technologies give rise to various kind of applications in real life scenarios. One prominent among them is the mid air user interfaces, where the use of hand and finger movements are used to control the interface. Ultrasound haptic technology [ITS08] is one of the example of contactless haptic technology where ultrasound waves are emitted by the device and coincide at a particular point in space to give haptic sensations.

1.1 About Ultrahaptics

The human skin has mechanoreceptors which are sensory cells that convert mechanical forces into nerve excitation. When the sound waves make contact with the skin, it gives a tactile sensation for the user. In order to produce such ultrasound waves we make use of a device manufactured by Ultraleap [Ul] called as Ultrahaptics. The device emits ultrasound from its phased array which propagates as a wave with a frequency higher than the limit of human hearing. Each emitter in the device emits the wave and every wave emitted coincide at a point in the air. This coincidence of the waves at a point is called a focal point and this focal point when directed towards a user palm creates a pressure for the user and the user can perceive it as a tactile sensation. The coincidence of waves at a focal point, strength and intensity of each transducer is handled by the Ultrahaptics device itself by using various solver algorithms.

In order to render custom shapes we need to explore various techniques of rendering and experiment with different parameters of the device. The different parameters available in the device are intensity, frequency, multiple focal points and the different techniques of rendering are explained in the next section of literature survey.

1.2 Project objective

Using the concept of mid air interfaces, we can explore and build many applications in our real world. In the recent times due to the COVID-19 pandemic, one prominent use case would be to build contactless interfaces in the public space. In order to achieve this we need to render custom shapes to the user in order to make user aware of different kinds of sensation. In this project, our main focus is to render custom shapes on user's palm using different kinds of inputs. Here, different kinds of inputs means different various formats like CSV (comma-separated values), SVG (Scalable Vector Graphics), user drawing and Leap Motion. We process all these kind of inputs and render the dynamic shapes using the Ultrahaptics device.

2 Related Work

> *A picture is worth a thousand words. An interface is worth a thousand pictures.*
>
> **— Ben Shneiderman**
> (Professor for Computer Science)

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

This is the second paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

2.1 Related Work Section 1

And after the second paragraph follows the third paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of

the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

After this fourth paragraph, we start a new paragraph sequence. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

2.2 Related Work Section 2

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A

blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

This is the second paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

And after the second paragraph follows the third paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

2.3 Related Work Section 3

After this fourth paragraph, we start a new paragraph sequence. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

This is the second paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the

alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

And after the second paragraph follows the third paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

2.4 Conclusion

After this fourth paragraph, we start a new paragraph sequence. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

3 Features

> *A good end product is not just a collection of features, Its how it all works together.*
>
> **— Tim Cook**
> -CEO Apple Inc.

As mentioned previously the main objective of the application is to consume multiple types of inputs from user and render it to the Ultrahaptics device. The features are built keeping those multiple inputs in mind. A display of multiple inputs can be seen in the image below. The features that are supported in the application are

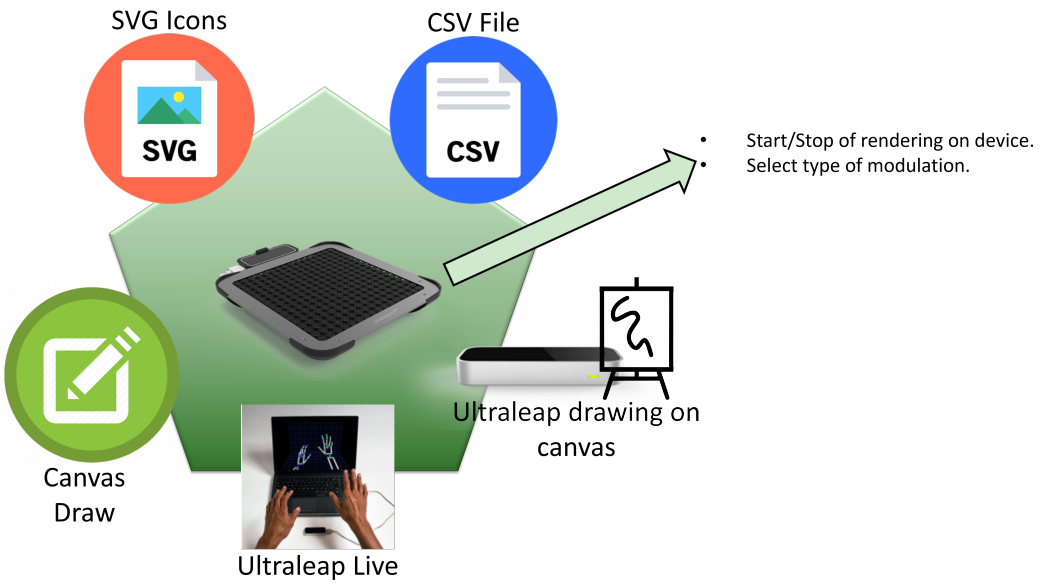


Fig. 3.1: Figure: Features of the application

categorized in three part:

a) Output: Ultrahaptics rendering.
b) Inputs: multiple types of inputs supported.
c) Controls: how user can control rendering.

and are explained in brief further on.

3.1 Output

Since this application revolves around one major output, it is important to mention the output in the beginning. The single output of the application is the custom dynamic shape to be rendered/emitted by the ultrahaptics device. Since, the device has limited size each input coordinate/s should be scaled accordingly.

3.2 Inputs

Multiple types of inputs are accepted in the application and tabs are created for different inputs, these tabs can be seen in fig 3.2:

Fig. 3.2: Figure: Tabs in WebApp

CSV File

A CSV file which has the coordinates of points in an x,y plane in column family. It can be uploaded in File tab. It assumed that the coordinates are mentioned in meters. But since the size of ultrahaptics is 16mm $\times$ 16mm the values in CSV are expected to be in that range. For eg the coordinate (4,5) mm should be 0.004, 0.005 in CSV file. An example of the csv input plotted on an xy plane can be seen in figure 3.2 only for the visual purpose.

In other forms of inputs to the application, the coordinates are scaled and centered

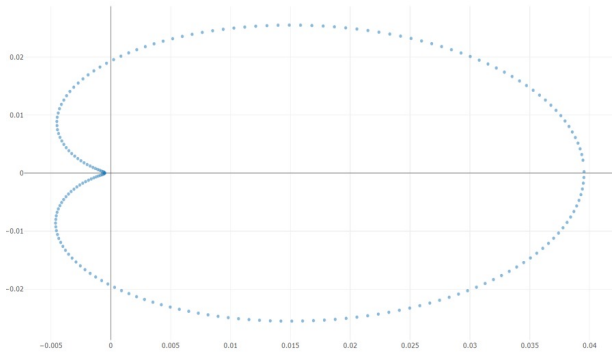


Fig. 3.3: Figure: example csv Plotted on xy plain

to size of the Ultrahaptics device by the application. However in csv input, it is assumed that the coordinates are scaled and centered. As soon as the user clicks on

the render button the coordinates are transferred to the Ultrahaptics device and it starts rendering.

SVG File

An SVG is also accepted as input format in the same tab as csv. As soon as the user uploads the SVG its thumbnail is shown on the same page to confirm the visibility. SVG coordinates are then communicated to the ultrahaptics device for that shape to be emitted after user clicks on render button.

Canvas Draw

This feature can be accessed under Canvas tab. A HTML canvas is provided in which user can draw basic shapes and patterns. The coordinates from the canvas is then communicated to the backend and rendered on the ultrahaptics device.

Ultraleap Live

Ultraleap[ultraleap.com] is an advanced hand tracking hardware sensory device that accepts hand and finger gestures as input, similar to a mouse, but without the need for physical contact. It is is a tiny USB peripheral device that is meant to be put on a physical desktop with its face forward. It's also compatible with virtual reality headsets. The gadget observes a roughly hemispheric region to a distance of about 1 meter using two monochromatic infrared cameras and three infrared LEDs[Wei+13]. The LEDs emit patternless infrared light, and the cameras capture almost 200 frames per second of reflected data[Lea]. This is then transferred to the connected computer via USB connection, where it is processed by the company's proprietary hidden code, synthesizing 3D position data by comparing the 2D frames recorded by the sensors.

Fig. 3.4: Figure: Leap motion controller

We consume the data provided by the leap controller and mimic the motion, movement and direction of moving index finger on the ultrahaptics board. another requirement of the feature is also an ultraleap device connected and setup on the system where the webpage is accessed. This feature is accessible in the tab leap live.

Ultraleap

Similar to the ultraleap live user can draw shapes on a canvas provided in WebApp using the leap motion controller. As soon as the user clicks on render button the shape is rendered on ultrahaptics board.

Note: To start tracking the index finger by the device on WebApp. First click on the Start Leap button provided in our WebApp and then do the click/tap gesture by your index finger above Ultraleap motion controller device. After that you will get a banner notification that the system is tracking your finger. Do the similar gesture to stop the tracking.

3.3 Controls

Some basic controls are also provided in the applications where user can secect and control other factors, such as:

Stop button:

On top right of our WebApp a stop button is provided, the rendering on the ultrahaptics device can be stopped anytime using this button.

Selecting type of rendering:

On top left of the webapp there is a dropdown select. Here user can switch between Amplitude Modulation or Time Point streaming for rendering. Once user has selected AM, and now wants to observe TPS, rendering has to be stopped by the stop button before switching.

4 Architecture

> *Any application that can be written in JavaScript, will eventually be written in Javascript.*
>
> — **Jeff Atwood**
> Code Horror (blog)

This section describes the usage and mapping of hardware and software of this application. This section will clear out some questions about the application like. How the application is consuming the inputs? How the application is transforming the data states to finally in the output? How are the middle layers processing the data? including some other questions about communication of different components. The aim of this section is to explain in detail the architecture of the application and flow of features. This will also explain the requirements to execute the application on local system.

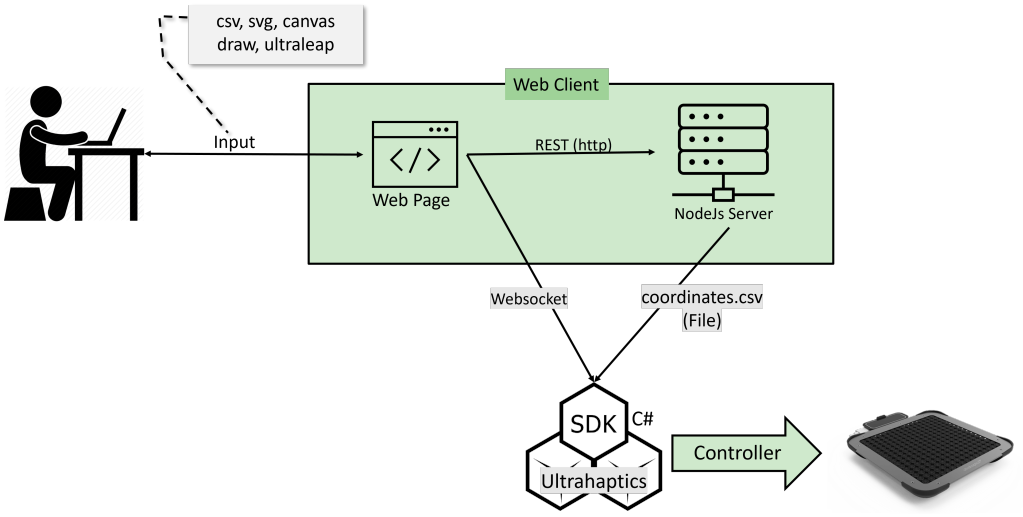


Fig. 4.1: Figure: The Haptic Printer (Application architecture)

As seen in the architectural diagram(Fig 4.1) there are multiple components of the application, from user input to ultrahaptics device to the communication through websocket and REST. The components are mentioned in detail below:

4.1 Frontend

The front end of the application is the web. The web application is built in vanilla-JS served over a nodeJs Server. However, for compiling js code for browser babeljs[Bab] is used.

The WebApp and Inputs

The webpage is responsible for taking multiple inputs from the user and start and stop Ultrahaptics device. The inputs are in form of the canvas draw, SVG icons, CSV file with (x, y) coordinates, Ultraleap inputs. The web page is responsible to scale and center the coordinates (according to x/y plane) taken from multiple input types to the Ultrahaptics dimensions and units.
In case of SVG some more processing is required. Since, there is no direct feature provided by the browser to extract the coordinates from an SVG. The code written for this is a custom implementation inspired from a spotify tool[Spo] which is under Apache license 2.0 . After extracting the coordinates from the given input the coordinates are scaled according to the output device and sent to the backend server in NodeJs, the processing in the nodeJs is explained in the section backend.
After, the successful response from the Node Server the front end is also responsible to communicate to the Ultrahaptics SDK implementation to start and stop the Ultrahaptics rendering.

4.2 Backend

The Backend is further divided into two parts. The First, is the Node Js Server. Second is, the system implementation in C-sharp, which is responsible to control and render shapes in Ultrahaptics device.

NodeJs Server

The functionality of nodeJS server is minimal. It hosts the WebApp at localhost:3000 in its root and also consumes the scaled coordinates from the WebApp. After getting the coordinates it creates a CSV file and writes all the coordinates in a csv as x,y as two columns. Post this functionality it responds to the webapp with a success message. This CSV file is later consumed by the SDK API implementation to render it on Ultrahaptics device.

The SDK extension: Ultrahaptics API's

Ultrahaptics SDK[Ultb] has provided some of the classes to manipulate the control points, intensity and how the emitter will respond to the control points and positions provided to it. We built a system using these API's to continuously render custom points and shapes provided. All the previous implementations and tests we found were for a shapes which were pre defined in the code and it was a challenge to dynamically change the emitter points at a given time.
This part of the system implements two types of rendering Amplitude Modulation(AM) and Time Point Streaming(TPS). Both the implementations consumes dynamic inputs and detail explanation of how it is implemented can bee seen below in code documentation in Files AM.cs and TPS.cs. The choice of rendering type between AM and TPS can be selected from the WebApp.
The system also implements a websocket for communication which will be discussed further.

4.3 Communication

Websocket

The websocket communication happens between the web app and Csharp Ultrahaptics system to consume API's. Where Csharp implements the websocket server $(forreference, seefileUltrahapticsOrchestrator.cs)$ and webapp implements the websocket client $(refseecswebsocket.jsbelow)$. The webapp sends a message to backend system to start or stop the Ultrahaptics device rendering, using websocket. It also sends what type of rendering to use(AM or TPS). the websocket communication message is in JSON format and an example message can be seen below.

```
{
  "action":"start",
  "type":"AM"
}
```

Fig. 4.2: Figure: Json example to start AM rendering

REST

The REST communication in between the webapp and the NodeJs server. After the input processing in the webapp the coordinates are sent to the NodeJs usin this REST channel at $localhost : 3000$.

CSV file I/O

After the NodeJs server receives the coordinates, the coordinates are written to a csv file in the column format. Column names (x,y). These coordinates file is picked up

5 Validation

After the development of our web application it was time for validation to check how the custom shapes are being rendered from the Ultrahaptics device. In order to validate our rendered shapes we shortlisted three different techniques which are explained in detail in the following sections.

5.1 Ultraviz tool

The Ultraviz is a tool developed by Ultraleap [Ul][Ulta] to visualize control points that are being rendered by the Ultrahaptics device in a three dimensional space. It helps the developer to understand how the rendering is happening visually. With the help of this tool we tried to analyse the different rendering techniques like AM and TPS with different kind of inputs i.e, CSV, SVG, drawing on canvas, live leap motion and drawing on canvas with the help of leap motion.

The following are the various kinds of inputs we tried to render:

Note: Since the images are screenshots of the Ultraviz visualization tool, we cannot see all the control points being rendered in the image as it is moving at a high speed.

1. Read CSV file feature:

The x and y coordinate values are read from a CSV file and the plot is as shown in Fig 5.1. This plot will be rendered in the Ultrahaptics device. Fig 5.2 (a) represents the CSV plot being rendered on Ultrahaptics using AM technique and Fig 5.2 (b) represents the same CSV plot in TPS technique.

2. SVG file input feature:

Fig 5.3 depicts the various SVG files used to render on Ultrahaptics device. Once the SVG files are uploaded it is converted to x and y coordinate system with many points. Fig 5.4 shows the different xy plot of the SVG files uploaded.

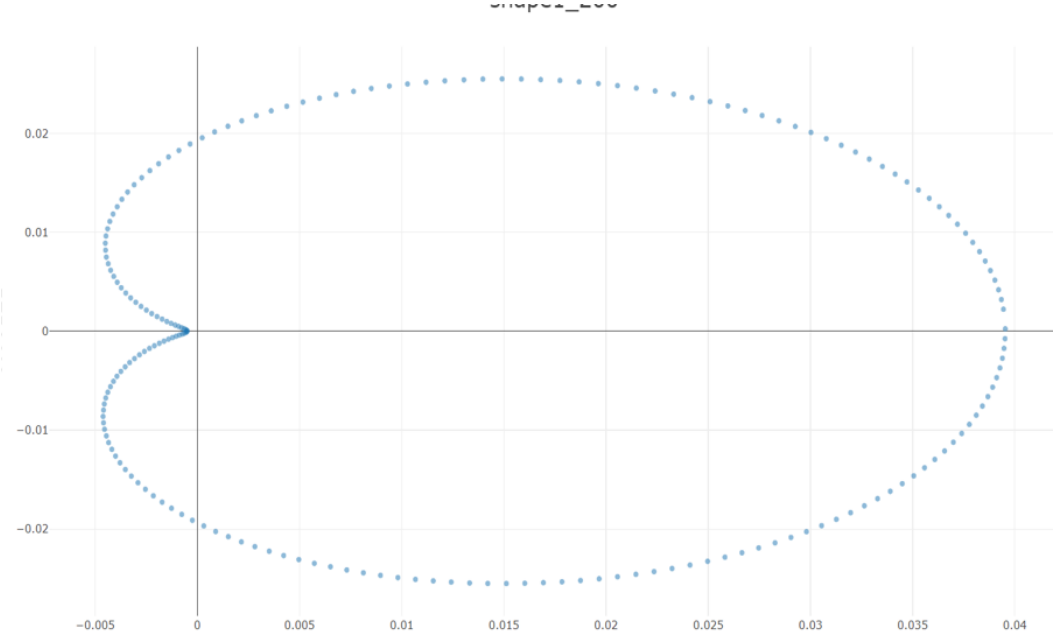


Fig. 5.1: CSV Plot of a custom shape

Fig 5.5, Fig 5.6, Fig 5.7 displays an image of Ultraviz tool rendering the SVG inputs arrow down, WiFi and square in AM and TPS technique respectively.

3. Drawing on canvas:

The user can draw the shapes on the given HTML canvas using mouse. Fig 5.8 (a) shows a user drawn circle shape and Fig 5.8 (b) shows an plot of the user shape represented in x and y coordinate which will be eventually rendered on Ultrahatics.

Fig 5.10 (a) depicts the rendering of custom user shape in Ultraviz in AM and Fig 5.10 (b) depicts the same in TPS technique.

4. Leap Canvas:

Leap canvas is similar to canvas drawing, instead of using mouse as a drawing tool here we use our index finger and with the help of Ultraleap leap motion device we draw on the canvas. Later the canvas drawing is converted to xy plot and then rendered on the Ultrahaptics device.

5. Leap Live:

Leap live uses the Ultraleap leap motion device to track our index finger and the same point is being rendered live on the Ultrahaptics device. The control point moves in the direction of the Ultraleap leap motion hand input.

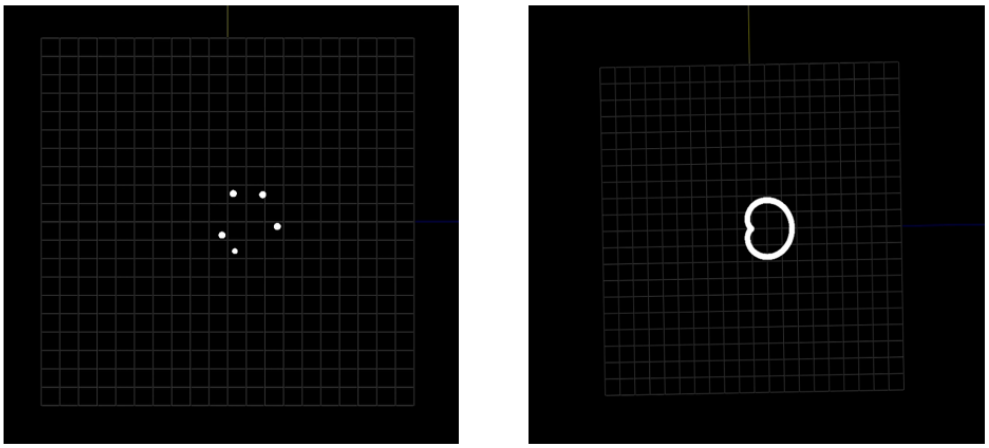


Fig. 5.2: (a) Ultraviz visualization of CSV plot in AM and (b) TPS technique

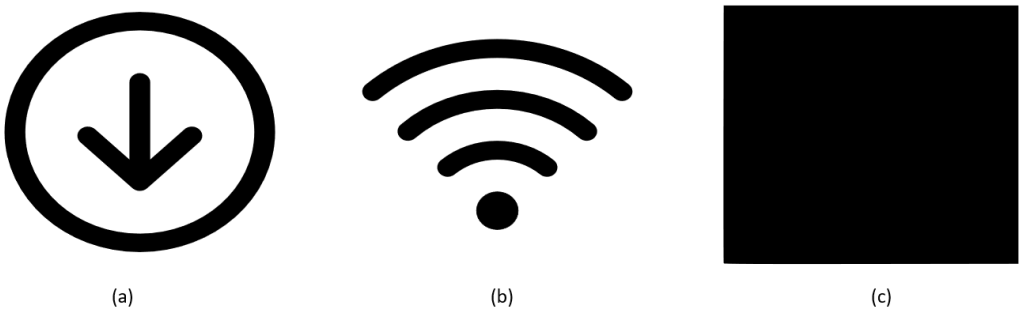
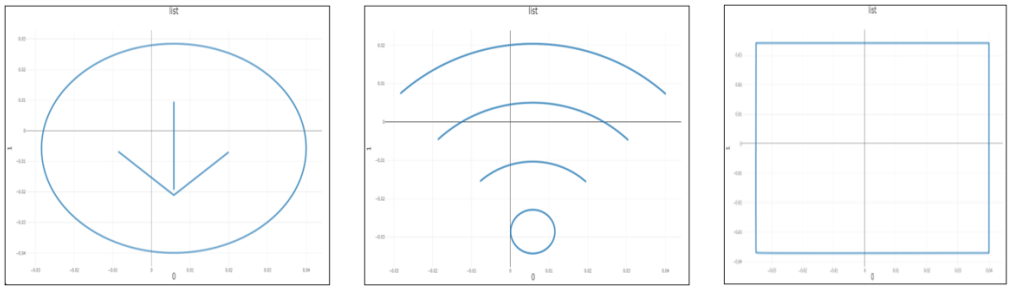


Fig. 5.3: SVG input shapes (a) Arrow Down (b) WiFi (c) Square

5.2 Cross validation and observation from developers

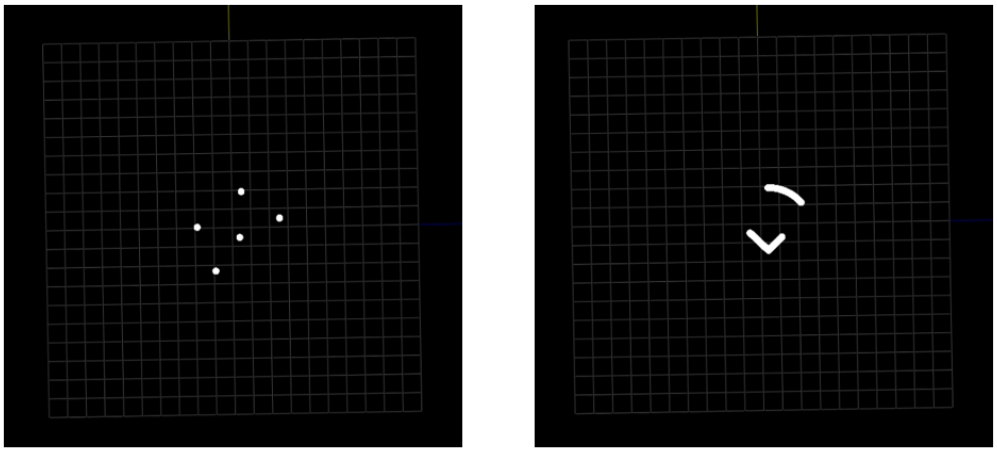
In order to further validate our project we conducted a short study between the authors of this project where we chose three different shapes i.e, square, circle and triangle and decided to guess the shape in various different techniques. In the first study we did not tell the observer on the Ultrahaptics which all shapes where there in the study and the observer guessed everything as circle.

Later in the second study, we informed the observer the available shapes and then performed the study. We conducted the study mutually between us and below are the results: Square shape was easily identified by both of us in AM technique and both guessed the square shape as circle in TPS technique. Triangle was guessed correctly in both the techniques. Circle was guessed correctly in TPS technique and one observer reported circle as triangle in AM technique.



(a) (b) (c)

Fig. 5.4: SVG plot of input shapes (a) Arrow Down (b) WiFi (c) Square



(a) (b)

Fig. 5.5: (a) SVG rendering of arrow down in AM (b) SVG rendering of arrow down in TPS

Below in Fig you can see all the results summed up in a table.

5.3 Oil bath

We also came across a technique where the rendering can be projected on a oil bath to see how the inputs where being rendered on the user palm [Oil]. The step by step guide to set up the environment is as follows:

1. The Ultrahaptics device is suspended approximately 15cm above the oil bath using a robotic arm. There is no need to use the Leap Motion so the cradle can be separated.

2. The oil is between 2 and 5mm deep. There is a purpose built perspex tank of 30 centimetres for the experiment. The oil must have the correct consistency and

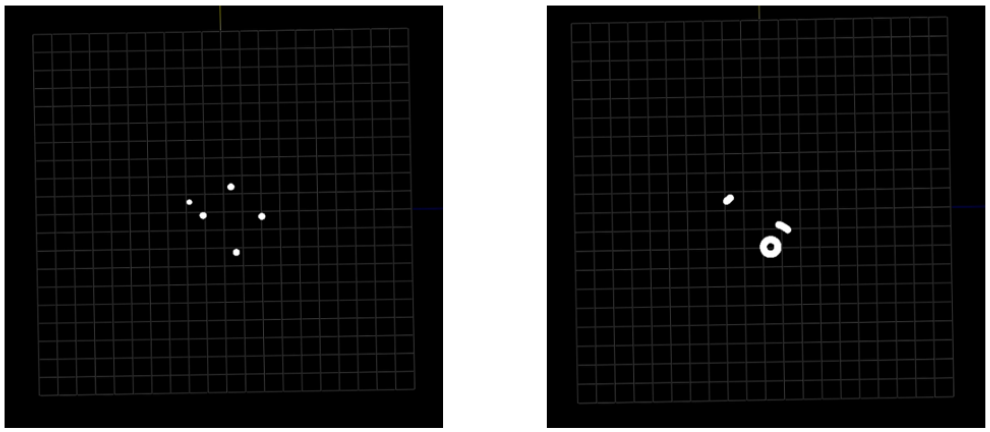


Fig. 5.6: (a) SVG rendering of WiFi in AM (b) SVG rendering of WiFi in TPS

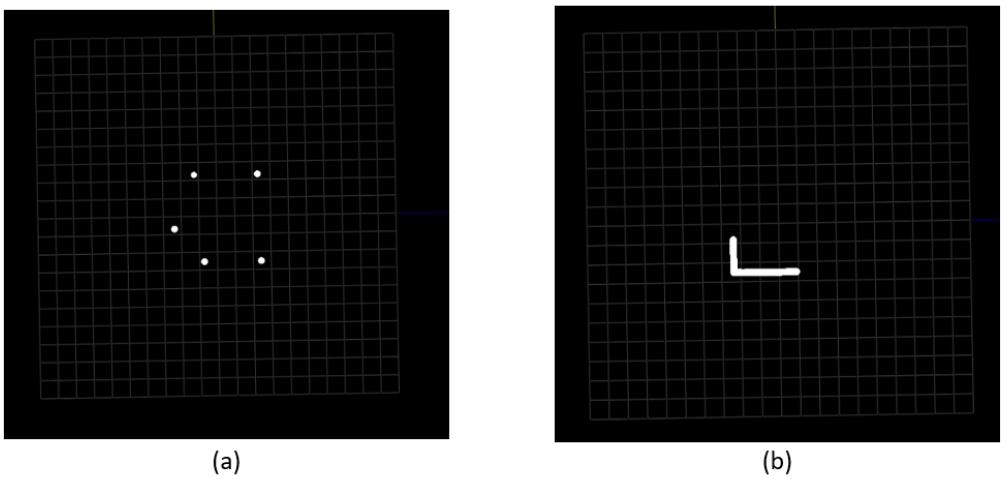


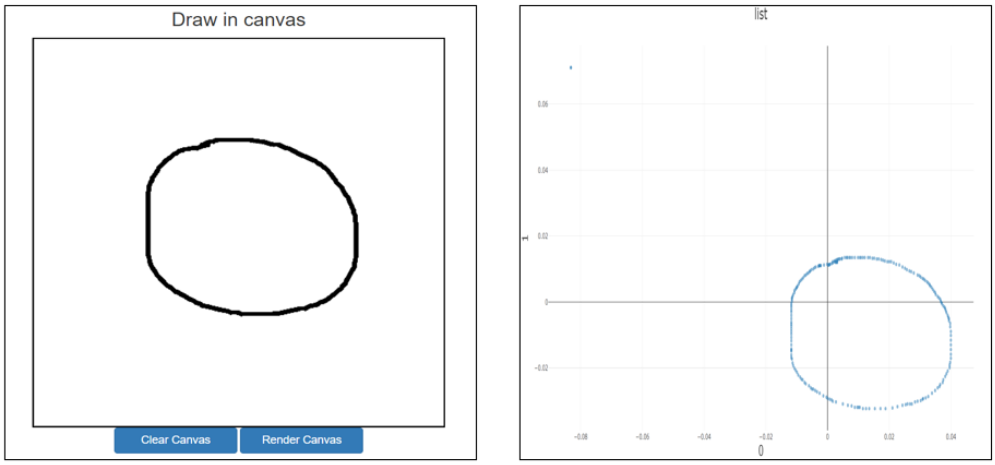
Fig. 5.7: (a) SVG rendering of square in AM (b) SVG rendering of square in TPS

viscous enough to show dispersion, fluid enough to be responsive. We have found that a 50:50 mix of olive oil and pumpkin seed oil give good results.

3. The oil bath must be raised approximately 3 cm above a white surface. We used legos for this purpose and used two A3 sheets of paper as a white surface.

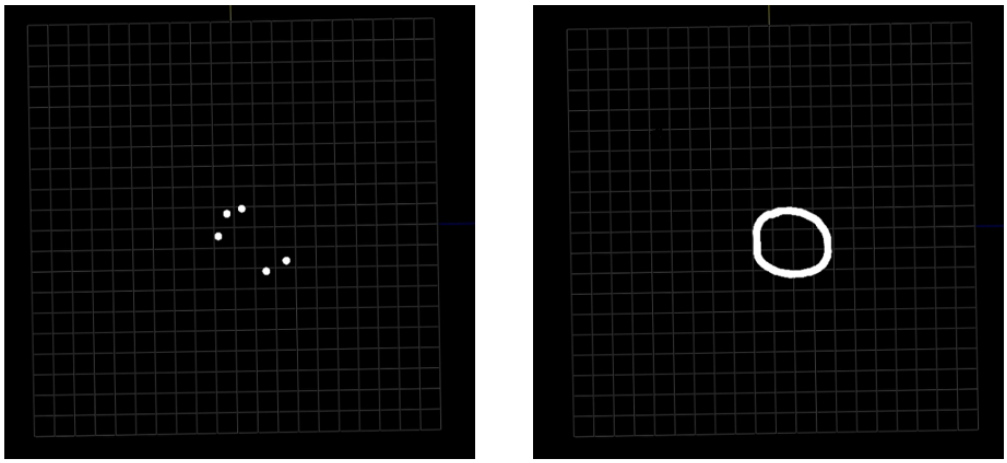
4. Light with a single, small, bright light source from above or to an angle to project the shadow of the distorted surface on to the oil. There is a small, LED light available on an adjustable stalk. Place it between top of tank and perspex holder, with light covering as much of oil as possible. The light must be bright enough to reflect onto a wall and the room must be as dark as possible to achieve the best affect.

Fig 5.11 and Fig 5.12 show the experiment setup of oil bath.



(a) (b)

Fig. 5.8: (a) User drawn circle shape on HTML canvas (b) XY plot of the given user shape



(a) (b)

Fig. 5.9: (a) User drawn circle rendering in AM technique (b) User drawn circle rendering in TPS technique

Observer: Anuj		
	Amplitude Modulation	Time Point Streaming
Shape		
Square	Yes	Answered as circle
Circle	Yes	Yes
Triangle	Yes	Yes

Observer: Akshaya		
	Amplitude Modulation	Time Point Streaming
Shape		
Square	Yes	Answered as circle
Circle	Answered as triangle	Yes
Triangle	Yes	Yes

Fig. 5.10: Results of validation between the authors of the project

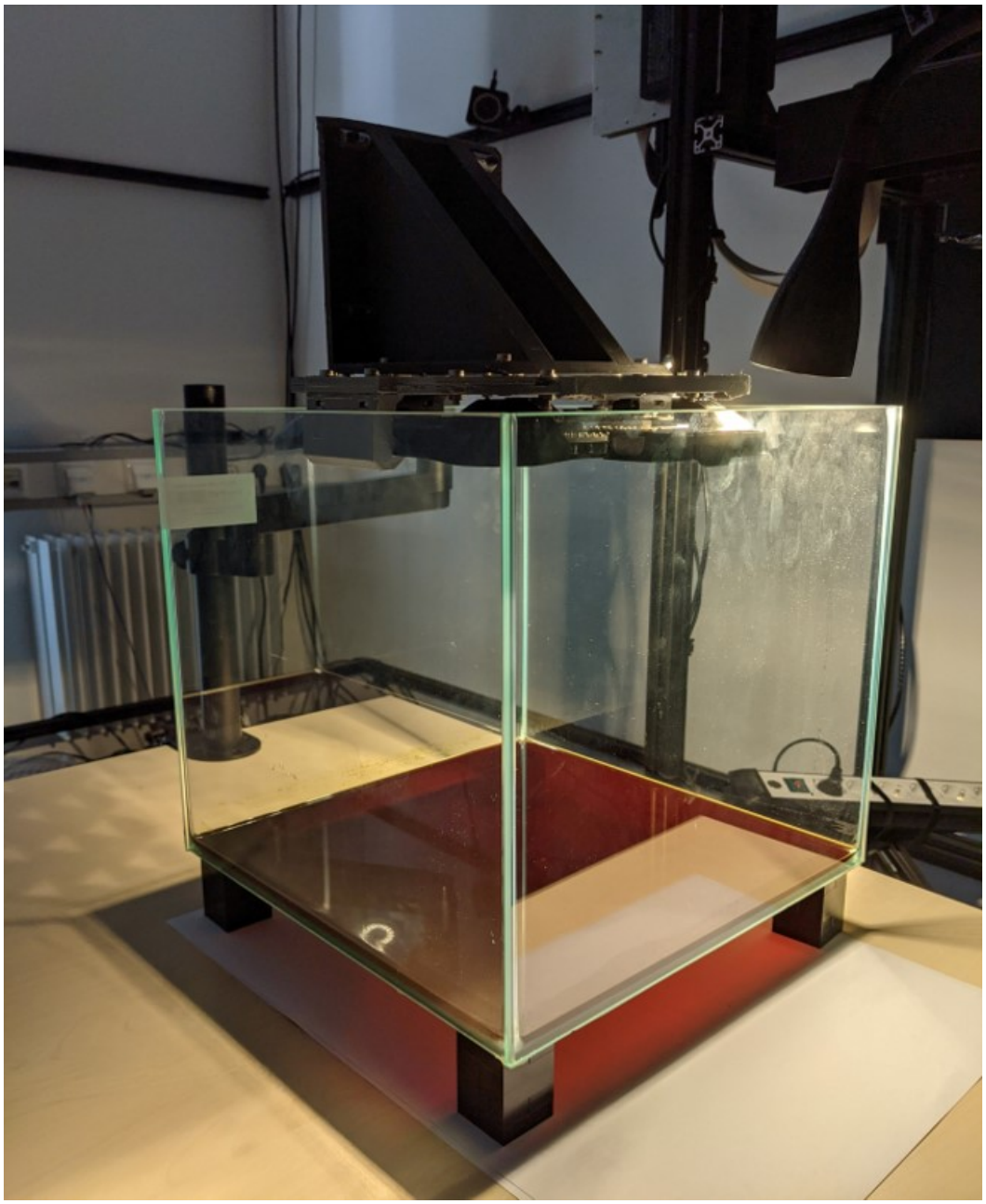


Fig. 5.11: Overall oil bath setup

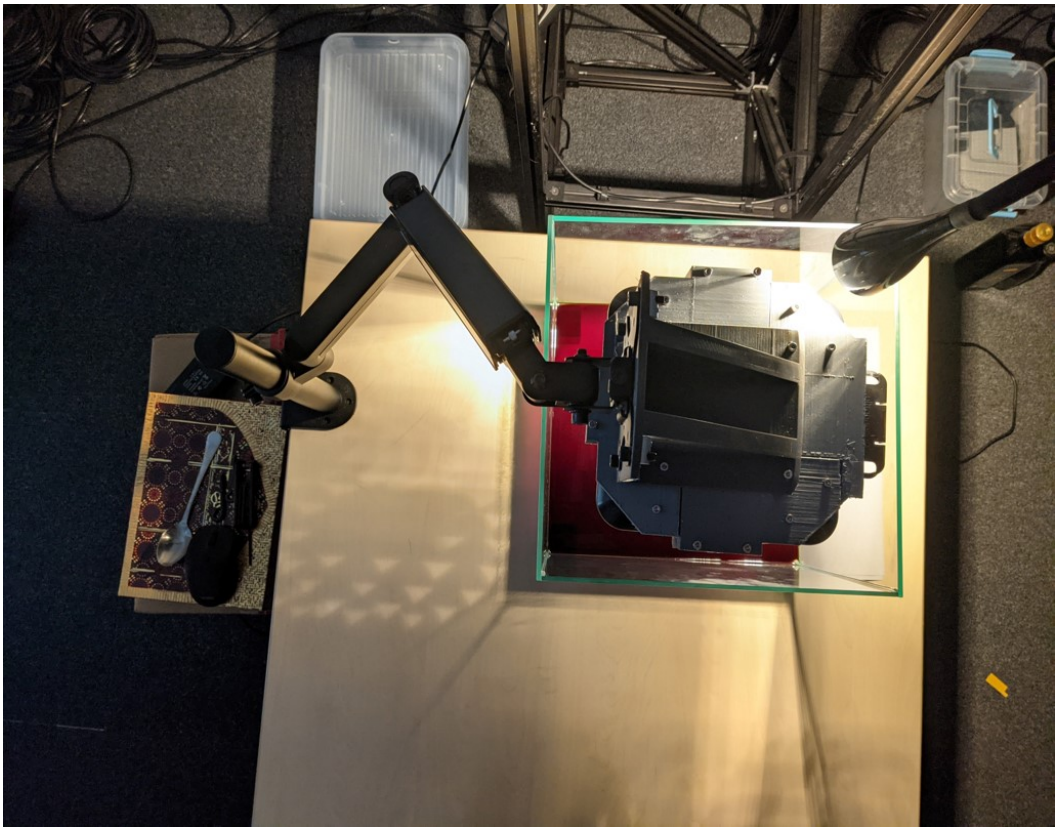


Fig. 5.12: Top view of oil bath setup

6 Conclusion

You can put your declaration here, to declare that you have completed your work solely and only with the help of the references you mentioned.

Bayreuth, September 27, 2021

The Haptic Printer

7 Future Work

You can put your declaration here, to declare that you have completed your work solely and only with the help of the references you mentioned.

Bayreuth, September 27, 2021

The Haptic Printer

Bibliography

[Bab] *Babel · The compiler for next generation JavaScript* (cit. on p. 14).

[ITS08] Takayuki Iwamoto, Mari Tatezono, and Hiroyuki Shinoda. *Non-contact method for producing tactile sensation using airborne ultrasound*. 2008, pp. 504–513 (cit. on p. 1).

[Lea] *Controller — Leap Motion JavaScript SDK v3.2 Beta documentation* (cit. on p. 11).

[Oil] *How does Ultrahaptics technology work? – Ultrahaptics Developer Information* (cit. on p. 20).

[Spo] *GitHub - spotify/coordinator: A visual interface for turning an SVG into XY coördinates.* (Cit. on p. 14).

[Ul] *Digital worlds that feel human | Ultraleap* (cit. on pp. 1, 17).

[Ulta] *ultraleap-labs/Ultraviz at master · ultraleap/ultraleap-labs · GitHub* (cit. on p. 17).

[Ultb] *Using Ultrahaptics APIs: Amplitude Modulation and Time Point Streaming – Ultrahaptics Developer Information* (cit. on p. 15).

[Wei+13] Frank Weichert, Daniel Bachmann, Bartholomäus Rudak, and Denis Fisseler. „Analysis of the Accuracy and Robustness of the Leap Motion Controller“. In: *Sensors* 13.5 (2013), pp. 6380–6393 (cit. on p. 11).

List of Figures

List of Tables

Declaration

I declare that this project, was solely undertaken by myself. All sections of the report that use or describe an concept developed by another author are referenced.

Bayreuth, September 27, 2021

The Haptic Printer