The University of Manchester
School of Computer Science

Third Year Project Report

3D POV (Persistence of Vision) Semi-Hologram

Aivaras Tumas

BSc Computer Systems Engineering

Supervised by Dr Paul Nutter

May 2017

Abstract

The aim of the project is to develop a system in the form of a display that would be able to draw 3D shapes hanging in midair that users have no interaction with – the semi-holograms. Given the design in a form of a 3D shape that users create, the system must be able to represent similar-looking objects that are without the need of using 3D glasses. Objects must have at least these two features that the real life objects do – they are three-dimensional and they look different depending on the point of observation. The importance of this project stems from the fact that there is a growing potential in 3D graphics industry and companies are looking for new ways of how to enhance the 3D visualisation experience.

Acknowledgements

I would like to express my gratitude and thank my supervisor Dr Paul Nutter for his continuous help and support during the process. He always kept me consulting, advising and inspiring by always setting higher standards than before and this is what kept me motivating during the development of this project.

Table of Contents

Glossary

2D – Two-Dimensional.

3D – Three-Dimensional.

POV – Persistence of Vision.

LED – Light-Emitting Diode.

FPS – Frames per Second.

Semi-Hologram – a three-dimensional object that looks different depending on the point of observation.

SD – Secure Digital (memory card format).

LCD - Liquid Crystal Display.

RPM – Rotations per Minute.

Horizontal Scaling – to add more nodes to (or remove nodes from) a system [16].

Vertical Scaling – to add resources to (or remove resources from) a single node in a system [16].

SRAM – Static Random Access Memory (memory type).

I/O – input/output.

GUI – Graphical User Interface.

Perspex – acrylic glass.

GND – ground pin (electronics).

DATA IN – data input pin (electronics).

DATA OUT – data output pin (electronics).

5V – five Volt pin (electronics).

ESC – Electronic Speed Control.

UI – User Interface.

DoS – Denial of Service.

Chapter 1 – Introduction

1.1 Motivation

The 2D POV display is an array of LEDs laid next to each other on a single layer and being spun at a high speed to achieve an effect of two-dimensional shapes being drawn in midair. The images are controlled by turning particular LEDs on and off at particular times (Fig 1). The image frequency (update rate) must be higher than a human eye image capturing frequency to be able to accomplish this optical illusion.

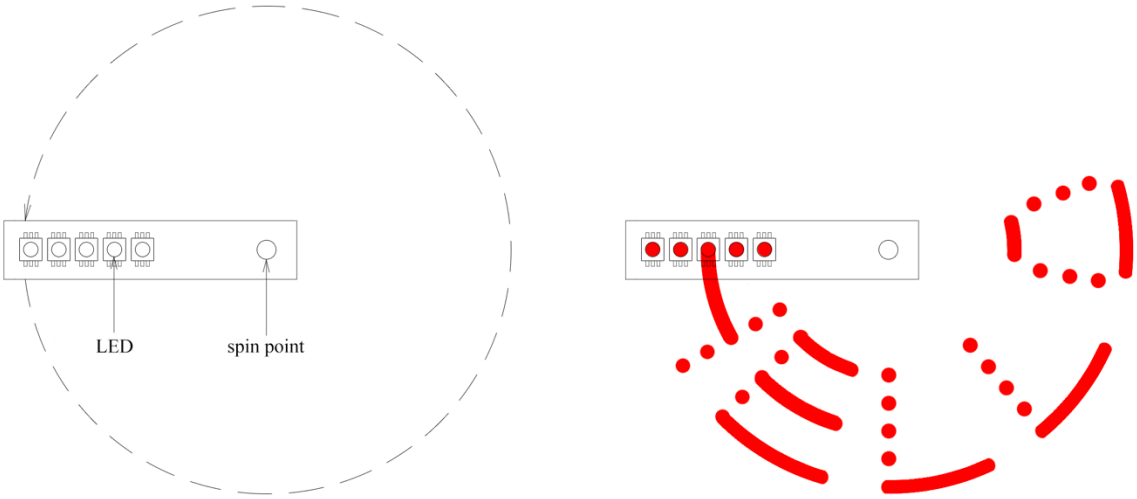

Fig. 1. A 2D POV display drawing “HELLO” by spinning counter-clockwise and turning LEDs on/off at particular times

A 3D POV display provides this technology with an additional coordinate – depth, so this way there is a possibility to draw more complex images that are three-dimensional. The simplest way to explain the difference between 2D and 3D shapes from the programming perspective is to understand that multiple 2D shapes put next to each other form 3D shapes. This is exactly the way how a 3D POV system works. The display consists of multiple layers stacked on top of each other that work the same way as the 2D ones, therefore, generating a 3D view to a viewer. However, this involves more programming to be done in order to synchronise the images between each layer and avoid image skew.

1.2 Current developments

The POV display technology has been under development by engineers for the last decade but with not many improvements on the user experience side of it which would be the basis for a fully-designed product that could potentially be sold. Looking at current developments of the 3D POV displays, we can see that there were not many approaches done towards this area of technology, with around 3 projects that could be found online [1, 2, 3]. Unfortunately, none of the products are being sold and have solely been done for research purposes,

therefore, there was no major concern of developing any good-looking, easy-to-use client software applications to create, save, update designs. Most importantly, none of the projects provide an ability to draw shapes at the center of the display ("spin point" in Fig 1) so there is no ability to draw shapes anywhere in the cylindrical space but only at particular places in that space.

1.3 Aims of the project

The summary of the goals of the study is below:

- Design a single proof-of-concept solution for a display that draws 3D images and provides an ability to do that anywhere inside the cylindrical space of the system, especially, in the center (Chapter 3).
- Gain general knowledge about how 3D image representation systems may be implemented and point out the current issues facing the technology (Chapter 4).
- Develop a simple light-weight client-side application in which users can easily create/update their own designs and interact with them (Chapter 4).
- An ambition to improve the current technologies of representing 3D images with no need of glasses to be worn and produce a fully finished product that could potentially be sold in the market (Chapter 6).
- Provide an analysis of the results of the project and provide an outline for further research (Chapter 6 and Chapter 7)

Chapter 2 – Background

2.1 POV

Persistence of Vision (POV) is the phenomenon whereby an after image remains on the retina of a human eye for a fraction of a second. This phenomenon is also believed to be how the human brain perceives motion [4]. The human brain combines multiple images together and this is how it understands that the object is moving. The simplest way to understand how it works is to look at what images and how they should be combined. For example, if multiple images of a horse rider were taken at different points in time (Fig 2) and were shown for some fraction of a second in sequence, starting from the top-left image and ending with the bottom-right image, a human brain would understand that as a movement – a human riding a horse.

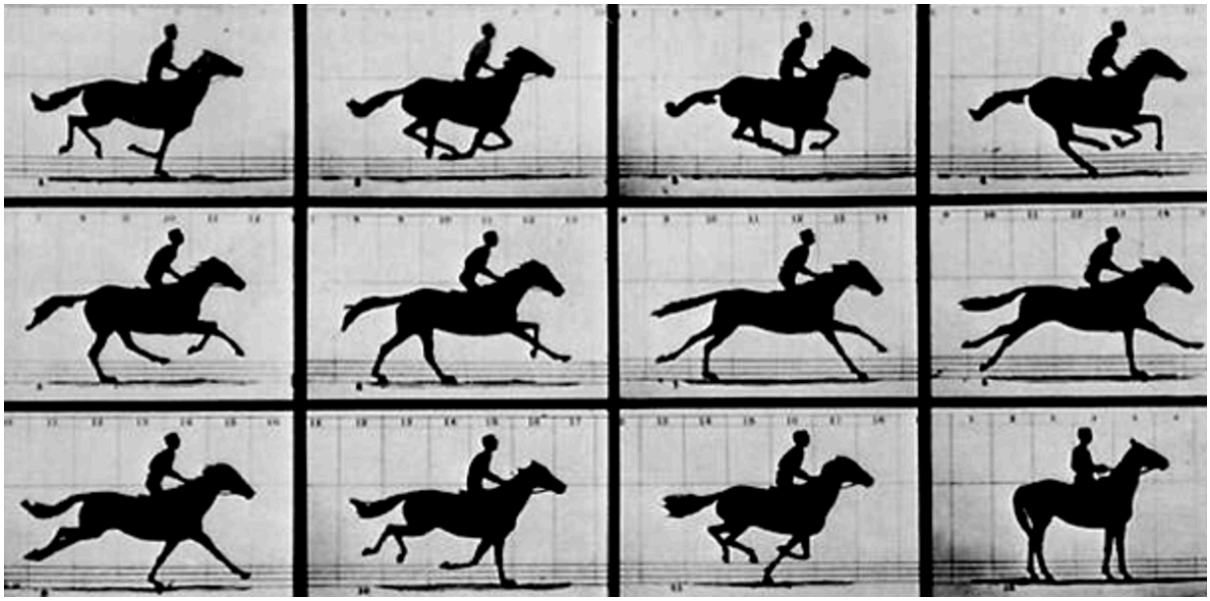

Fig. 2. Images of a horse rider taken at different points in time [5]

In short, a movie is a combination of different images (frames) that are shown at different points in time. Understanding the POV phenomenon has led to new advancements in cinematography and filmmaking. One of the first practical implementations of POV was the invention of a filmstrip, which was a “spooled roll of 35 mm positive film with approximately thirty to fifty images arranged in sequential order” [6] that were popular in the late 1960s and were used to project movies.

To create a sense of smooth motion within a movie, the frequency rate (the rate at which images change) must be equal to or higher than the rate at which a human eye captures images. “The human eye and its data reception and transmission system can form, transmit and analyse 10-12 images per second” [7]. This means that a visual system understands images individually, but if the rate is higher than 10-12 FPS (frames per second), it is then perceived as motion. “At slower rates of speeds (i.e. 16-24 times per second), the changes in brightness are perceived as flicker, which becomes more distracting the greater the brightness of flash” [7]. Therefore, the 24 FPS is high enough for humans to perceive motion with no disturbance.

2.2 3D image representation

Nowadays, most of the 3D graphics depend on 3D projection. It is a technique that maps three-dimensional objects into a two-dimensional plane. However, any two observers at any two locations (coordinates) see the exact same image (Fig 3 - Top). Therefore, the view does not depend on the location of a viewer. In contrast, holograms are 3D images that have a location dependency, so two observers at any two different locations would be seeing different sides of the same object (Fig 3 - Bottom). Therefore, the observers would be seeing different images depending on where the observers are in space.

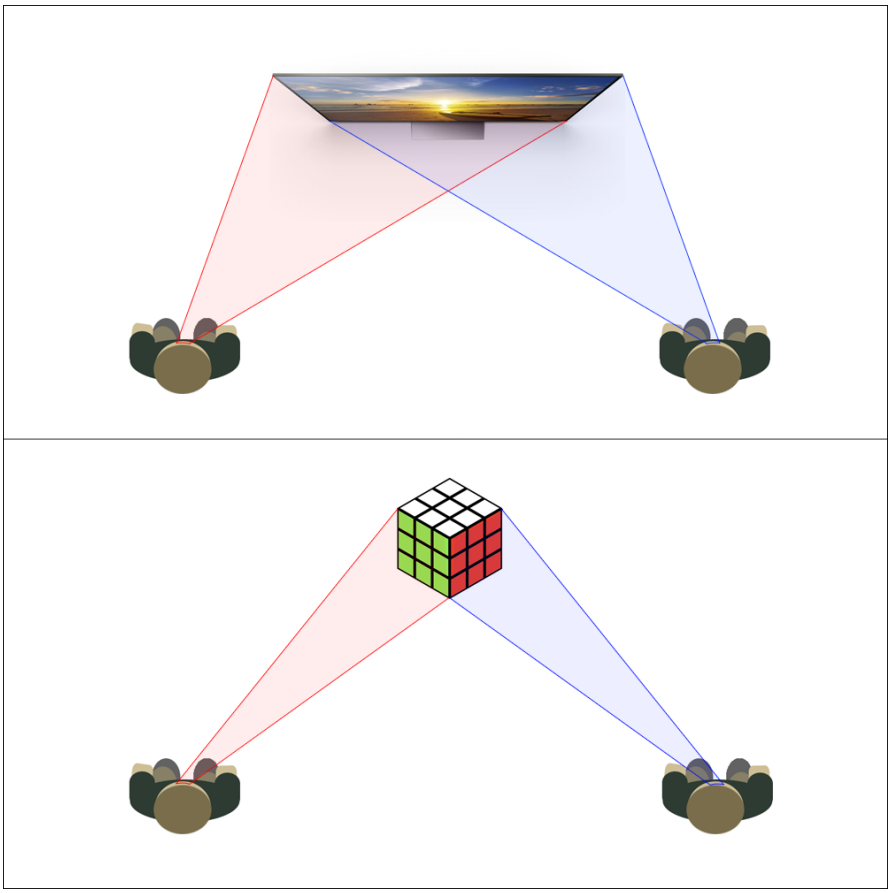

Fig. 3. Top image – a 2D image observed by viewers at different locations, Bottom image – a 3D hologram observed by viewers at different locations

Holograms have the same exact four features as the real-life objects do:

1. *They are three-dimensional*
2. *They look different depending on the point of observation*
3. *Possibility to change their location by directly interacting with them*
4. *Possibility to change their shape by directly interacting with them*

An important observation is that real-life holograms that obtain all of the four features have not been invented yet, however, some “semi-holograms” already exist. Semi-hologram is an unofficial name that will be used in this report to refer to a hologram that only has the first two features.

2.3 Semi-hologram technology

There currently exist three types of semi-holograms. The first type is a laser-based technology that points multiple lasers towards humid air, so the water droplets reflect the laser beams, therefore, drawing points in space that form 3D shapes [8]. The second type is a glass-based hologram technology that uses either one or four glass windows that partially reflect the light emitted from LCD screens that are placed above or below the windows [9]. The third type is a motor-based technology that spins multiple arrays of LEDs and turns those LEDs on and off at particular times, that way, creating an illusion of three-dimensional objects hanging in space [10].

Fig. 4. Three types of holograms. Top image – using lasers and mirrors [8], middle image – using LCD screens and glasses [9], bottom image – using LEDs and motors [10]

The main advantage of using a laser-based semi-hologram is an ability to draw high quality images. Some of the drawbacks are its high cost and complexity to implement. One of the major benefits of using glass-based semi-holograms is its low price, however, the images have a very low degree of opacity, so they would not be visible in a bright environment. The two most important advantages of using a motor-based semi-hologram system is that it is cheap to implement and works in both low-light and high-light conditions.

2.4 Market

The two most advanced products currently in the market are “viSio” and “voLumen”, both developed and released by a university student Maximilian Mali [11]. “viSio” claims to be the world’s first volumetric 3D display on the market that is capable of displaying quick-rate animations from an SD card. “voLumen” is a similar yet more advanced and stabilised volumetric display with the purpose to be used as a device that shows various three-dimensional advertisements.

2.5 Potential

Despite a low number of products currently in the market, there are new opportunities available for research in this area of technology. There are multiple areas in which this technology might be implemented and used. In medicine, holographic interferometry (analysis of structures with no damage to original samples) would lead to safe analysis, quicker and more in-depth 3D brain, molecule, DNA studies [12]. In art, holographic copies of important original art objects might be showcased or preserved, architectural models of buildings could be showcased [12]. In entertainment industry, this technology would provide an ability to play 3D games or watch 3D movies without the need of 3D glasses. In business, semi-holograms could improve the way employees communicate by showing holograms of people during meetings [12]. The points discussed above indicate that there is potential in this technology and various multiple areas might be positively influenced by it.

Chapter 3 - Design

3.1 Mechanical design

3.1.1 Requirements

There is a relatively huge variety of options on how to design the mechanical system of a semi-hologram (more than in either hardware or software designs), therefore, there must be clear requirements defined that would allow completing this project in a timely manner. The four functional requirements of the mechanical design are:

1. *The system has to work in any light conditions*
2. *An ability to draw shapes anywhere inside the cylindrical space of a rotational part of the device*
3. *There should be no or slight vibrations caused by the device*
4. *The microcontroller must not spin*

The three non-functional requirements are:

1. *The construction has to be inexpensive and affordable as a university project*
2. *The device has to be portable and easy to carry*
3. *The device has to be reusable with the current assets*

3.1.2 Laser-based, glass-based or motor-based?

As discussed in Chapter 2.3, there currently exist three possible ways to design a semi-hologram. The laser-based technology is too expensive and would require too much time to implement as a university project, however, this design would achieve the best performance. Furthermore, the glass-based semi-hologram technology is cheap but would not necessarily work in a bright environment. In contrast, a motor-based device is relatively cheap to implement and would work in any conditions. Therefore, based on the factors mentioned above a motor-based approach was used to design and develop a semi-hologram in this project.

3.1.3 Cylindrical space

One of the goals of the project was to design a device that is able to draw three-dimensional shapes anywhere inside the cylindrical space of the system. As opposed to other devices on the market (that all lose some cylindrical space at the center – Fig 1), this device has to have LEDs all over a layer, especially, at the center of the spin point. Therefore, there must be something holding the layer from the side although the spin point must still be the same.

3.1.4 Layer design

The simplest way to arrange all the LEDs together is to design a rectangular grid in which all the LEDs are placed on layers and each layer goes on top of each other. If looking from the side, the LEDs would look like a rectangle (Fig 5 - Left), however, if looking from the top, only the LEDs from the top layer would be visible (Fig 5 - Right), which consequently reduces the viewing angle significantly.

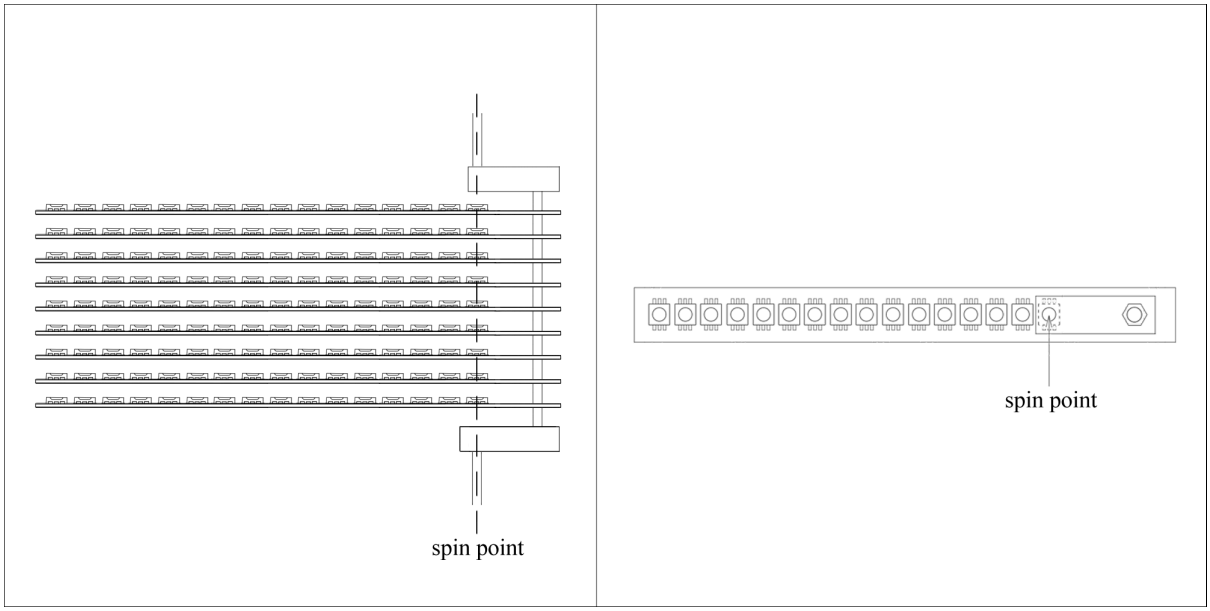

Fig. 5. Left image – side view of a grid of LEDs,
right image – top view of a grid of LEDs

In order to design a system in which the LEDs do not cover each other while spinning and allow a 180-degree viewing angle, the layers have to be rotated from each other. At the time of developing this project, the longest LED strip available online was the “Adafruit NeoPixel Digital RGB LED Strip 144 LED - 1m Black” [13] consisting of 144 LEDs. Therefore, the design has 9 layers each consisting of 16 LEDs which totals to 144. The simplest way to put the layers together is in a single-helix arrangement (Fig 6) with a 40-degree rotation angle

between each layer. Both designs (Fig 5 and Fig 6) have only single layers of LEDs at particular heights so there is no redundancy in LEDs which leads to a single refresh rate (instead of double or more). In addition, this design (Fig 6) allows to draw shapes anywhere inside the cylindrical space of the system.

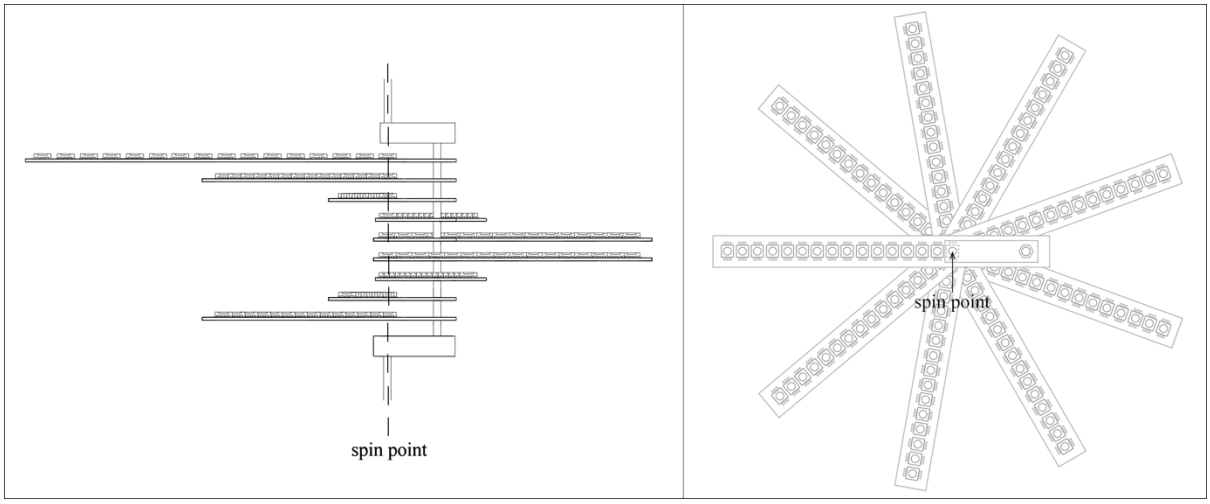

Fig. 6. Left image – side view of a single-helix LED arrangement,
right image – top view of a single-helix LED arrangement

Each LED can represent 45 pixels, with 5 pixels in between each layer. Therefore, the total number of pixels that can be represented in this system is 45 * 16 * 9 (6480), where “16” is the number of LEDs placed on each layer, and “9” represents the number of layers.

3.1.5 Overall design

To firmly hold the spinning layers there must be a strong metal frame built around them in order to achieve balance and avoid vibrations within the system. To simplify the design and reduce the workload of the motor, a pulley and a belt have to be used. A smaller (driver) pulley would be spun by the motor and a belt joining the two pulleys together would allow the larger (driven) pulley to be spun which would consequently spin the layers mounted on top of it. The signals from the microcontroller would be transferred by a slip-ring that would be placed on top of the frame (just above the layers), so the top wires would connect to a non-moving microcontroller and the bottom wires would connect to the layers that are being spun, therefore, allowing a static (non-moving) microcontroller pass the signals to a dynamic (moving) object. The RPM (rotations per minute) rate has to be tracked in order to calculate how long should an LED be on or off on each revolution. Thus, a Hall effect sensor needs to be placed at the bottom layer of the system and a magnet must be placed in front of the bottom layer.

3.2 Hardware design

3.2.1 Requirements

The focus of this project is to develop an abstraction that translates a user-defined 3D shape design from a client-side and converts that into an actual 3D image representation on the hardware side. Thus, there is no emphasis on designing a microcontroller on its own, and a premade microcontroller board can be used. On the hardware side, there are five functional

requirements that the microcontroller must meet:

1. *Send precisely timed signals to the LEDs*
2. *Keep track of RPMs*
3. *Control the motor*
4. *Be able to receive data from a client-side application*
5. *Signals must not interfere with each other, so different peripherals with different clock rates must not affect each other*

There are four non-functional requirements that the whole hardware system must meet:

1. *The design has to be nearly ready for deployment*
2. *The system has to support open source development*
3. *The design has to be reliable (time between failures maximised)*
4. *The design has to provide good developer usability (development time minimised)*

3.2.2 Microcontroller board and motor

One of the most common microcontrollers that provides an outstanding open source development is Arduino Uno. It has an ATmega328P chip with a 5V operating voltage [14] which is usually a standard and should be enough to power the LEDs. The microcontroller has a 16MHz clock speed [14] which is a high enough rate to precisely communicate with the peripherals. Arduino Uno has a built-in 32KB flash memory [14], with a 2KB SRAM [14]. The microcontroller provides 14 Digital I/O (input/output) Pins and 6 Analog Input Pins [14].

3.2.3 LEDs

As mentioned previously (Chapter 3.1.3), the Adafruit NeoPixel 144 LED strip will be used which will be cut into 9 pieces of 16 LEDs, then placed on the layers and soldered in series. The Adafruit NeoPixels were chosen because of their good compatibility with the Arduino Uno board (less wiring) and it also provides two Arduino libraries that might be used - *Adafruit_NeoPixel.h* and *FastLED.h* [13]. In addition, the LEDs can be addressed as a function below:

$$f(\alpha, \beta, \gamma, \delta) \tag{1}$$

In this context, α denotes the LED index (0-143) starting from the nearest connected LED to the microcontroller [15], β denotes the brightness of the red colour (0-255) of an LED [15], γ denotes the brightness of the green colour (0-255) of an LED [15], δ denotes the brightness of the blue colour (0-255) of an LED [15].

3.2.4 Scalability

The level of hardware scalability in this project is likely to be high, mainly because of the custom-made microcontroller board and a good support and compatibility of LEDs with the Arduino. If the microcontroller becomes faulty, it could easily be substituted with almost any Arduino board (Arduino Nano, Arduino Mega etc.) which only differ in size and the number of input/output pins. One of the main advantages in terms of opportunities to scale the hardware design is that Arduino-compatible shields can be plugged on top of the PCB board that provide extra features and new capabilities. Arduino boards being open source extend the level of scalability because of the support that could potentially be received online. The C programming language used in Arduino provides an ability to relatively easily develop hardware projects with minimal knowledge about electronics.

3.3 Software design

3.3.1 Requirements

In order to achieve maximum user experience (Chapter 1.3), there must be a client-side application developed which users interact with and this will be the only application that users have control of. The application has four functional requirements:

1. *Support for designs of three-dimensional objects*
2. *Users' ability to create, update, delete their own designs*
3. *Users' ability to rotate shapes, change colours of LEDs*
4. *Users' ability to login so different projects made by different users do not interfere*

The software-based application has five non-functional requirements:

1. *An application has to be lightweight, simple and easy to navigate*
2. *Relatively secure (private login credentials)*
3. *Provide good testability*
4. *Provide both horizontal and vertical scalability*
5. *Provide good maintainability (data stored on a server rather than locally)*

3.3.2 Mobile, desktop, or web-based?

There are three alternatives to choose from when deciding what platform should the software application be based on. One of the most important factors for choosing a platform is the level of support and the level of complexity in order to build 3D-based applications. The majority of mobile-based 3D libraries are mostly engines for developing games (such as *Unity* [17] or *LibGDX* [18]), however, there is a small number of libraries that have good support in developing three-dimensional applications. In addition, there is more pre-work to be done (e.g. setup how a user controls the application) in order to begin focusing on the core functionality of the application. There are significantly more libraries available that provide an ability to develop desktop-based applications with 3D support. However, most of them purely are heavy-weight 3D engines focusing on maximising the time performance and rendering (such as *jMonkeyEngine* [19] or *Java 3D* [20]). Furthermore, a front-end GUI (graphical user interface) must be developed in order to use any of the engines. Web-based software has significant amounts of support for creating 3D applications (such as *Three.js* [21] or *Babylon.js* [22]). One of the many advantages of developing a web-based application is that usually there is no need for users to download any additional software to their local machine in order to run the application because the application is stored on a server that clients only connect to. In addition, most of the libraries are light-weight that allow quick development of 3D-based applications.

3.3.3 Server and database

To allow users to login to the system, there must exist a database and a server in between the client-side software application and the hardware system of the semi-hologram (Fig 7). Web-based programming languages provide excellent support for sending queries to databases. The database would store users' usernames, encrypted passwords and all the data about projects. The server has two purposes - firstly, it will act as an interconnection between the client-side and the hardware system, thus, allowing encoded data to be sent about 3D shapes, secondly, server will be sending queries to the database about any changes within the application (e.g. size or colour of a 3D shape was changed).

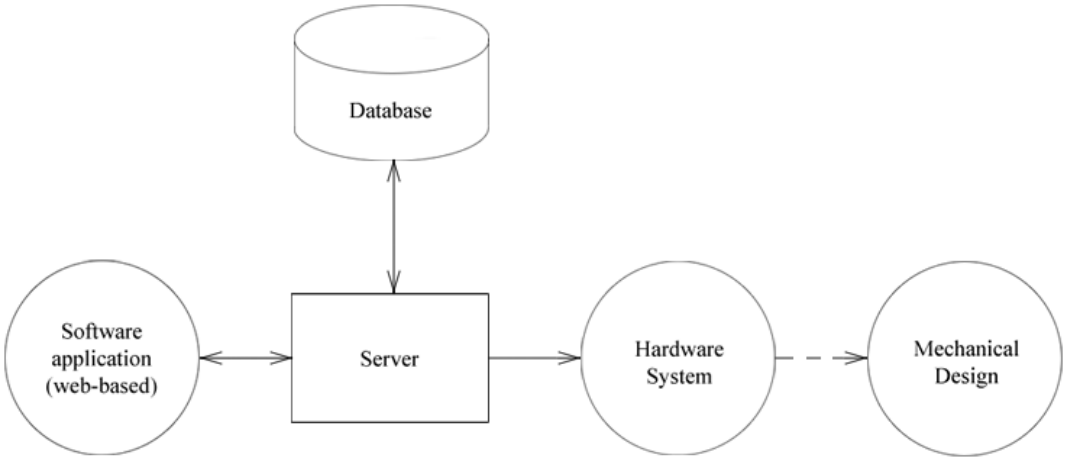

Fig. 7. Simplified view of the semi-hologram system. Two-way communication between the software application and a server, and between the server and the database. One-way communication between the server and the hardware system. The hardware system controls the mechanical design by sending electronic signals.

3.4 Interconnect protocol

In order to send data from the client application to the hardware system, a protocol must be defined. Client application will define data about pixels - information about the LEDs at specific points in time. To minimise transmission latency and an amount of data that gets sent, only the information about pixels that are on will be sent. To avoid confusion, in this report the word "pixels" will refer to the LEDs that are turned on at specific points in time. In addition, to minimise the amount of work on the hardware system – all the pixels will be encoded on the client-side and will be decoded on the hardware system according to the protocol. We shall now proceed to discussing the actual implementation process of the semi-hologram and what changes have been made to the aforementioned initial design.

Chapter 4 – Implementation

4.1 Mechanical design

4.1.1 Layer and LED design

Since the mechanical design of the semi-hologram was discussed in Chapter 3.1, this chapter will mainly discuss the approach of how the design was practically implemented. The first stage of implementing such a mechanical system was to decide how the pixels should be drawn by the device. As mentioned previously (Chapter 3.1.4), there is a 40-degree angle between each layer (Fig 8 - Left) and there are five pixels (arbitrarily chosen) in-between for each LED. An important observation is that the LED at the center spins around its axis, so its coordinate will never change, thus, it can only represent a single pixel (Fig 8 - Right). In addition, pixels that are further away from the spin point automatically have a longer width depending on the distance from the center (spin point). Small gaps exist between pixel lines

caused by the distances between each LED (Fig 8 - Right).

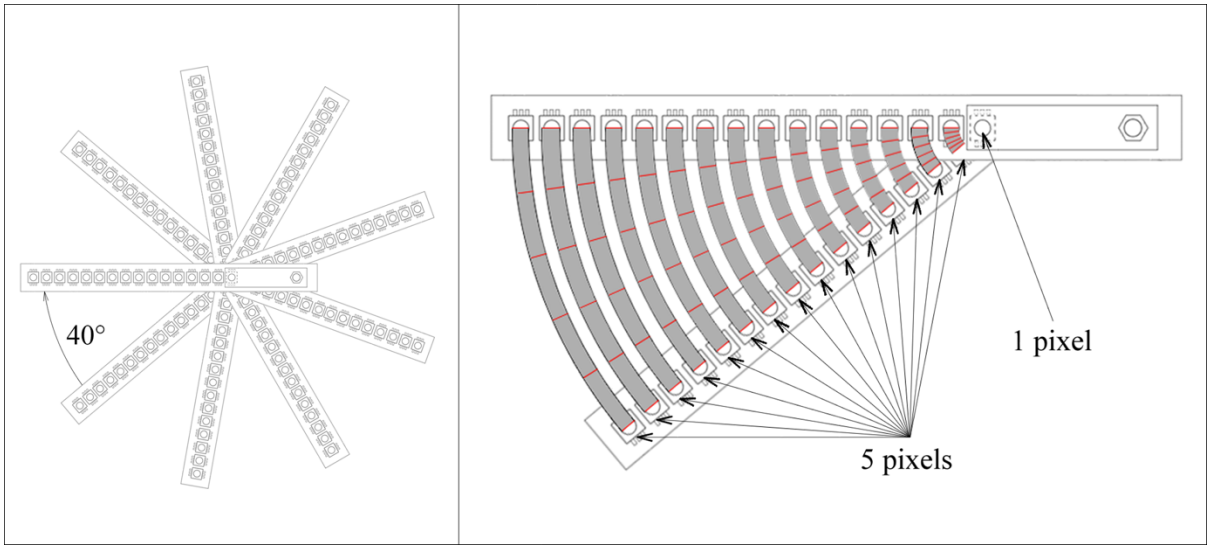

Fig. 8. Left image – there is a 40° angle between each layer, right image – there are 5 pixels (in gray, split by red lines) between each layer for each LED, the LED at the center has only 1 pixel

To draw a single line within a layer, a simple line must firstly be drawn on top of all the pixel boxes, which consequently gives a better look of where the collisions may occur between the line and the pixels (Fig 9 – Left). Then, the collision detection mechanism may be run (Chapter 4.7.2) in order to decide which pixels should be turned on. The line will look split and skewed (Fig 9 – Right), however, minimising the LED diameters and distances between the LEDs would maximise the quality of images being drawn.

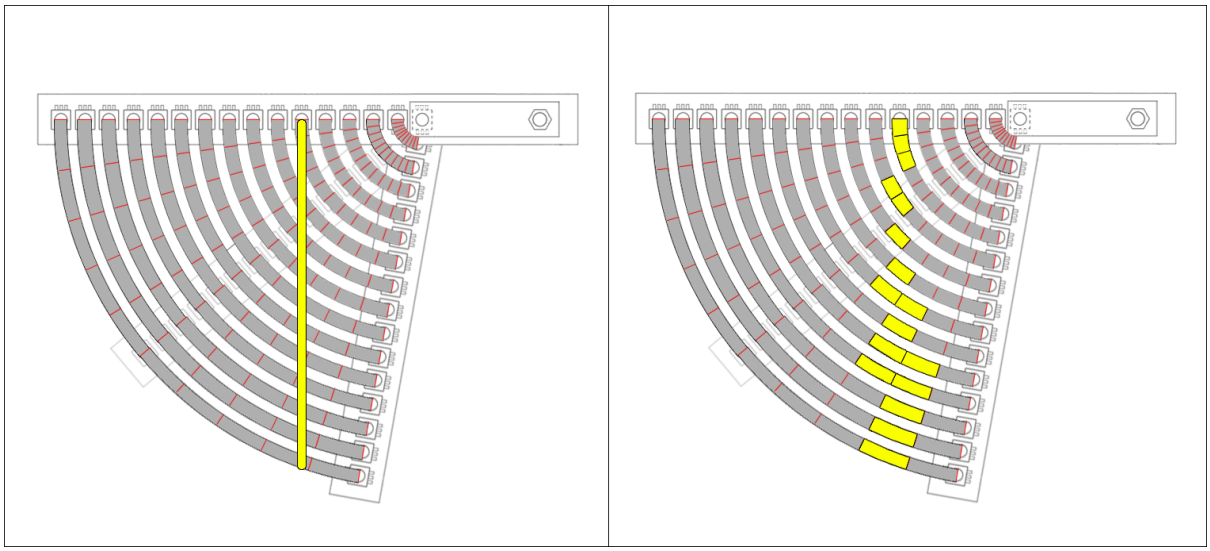

Fig. 9. Left image – representation of how a single yellow line looks in reality,
right image – representation of how a single yellow line would look in a semi-hologram device

The second stage was making the mechanical design. The layers that form the single-helix arrangement were laser cut from a 3mm thick Perspex sheet (Fig 10). Every layer has a length of 227.6mm, 15mm width, with a 55.7mm arm pointing out at different angles depending on which layer it is. The top layer is straight and has no arm, the second from the top layer has an arm pointing at 40-degree angle, third from the top layer has an arm pointing at 80-degree angle and so on.

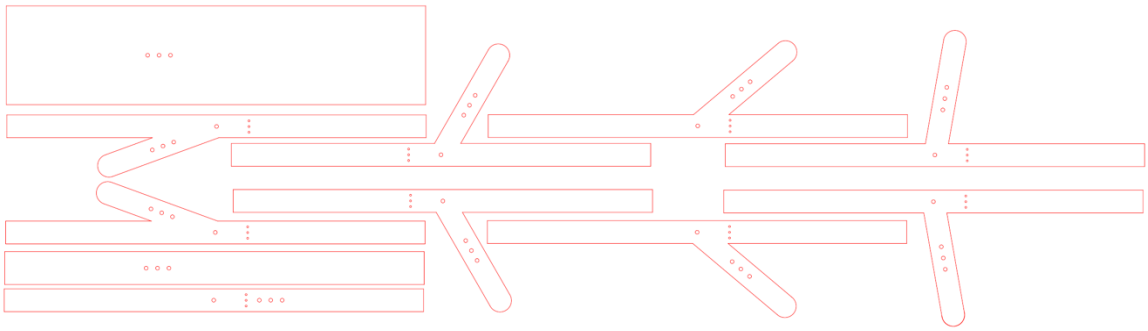

Fig. 10. Layer design ready for laser cutting. The top layer is shown at the bottom left

The width of the Adafruit Neopixel strip was 15mm, whereas the total length was 102.24cm. When cut into 9 pieces, the length of each smaller strip was 113.6mm, which nicely fitted on one side of each layer and the first LED of each strip would always be on the spin point.

4.1.2 Mechanical parts

The frame used was made of metal in order to maximise the stability of the design and minimise the amount of vibrations. The metal frame was holding the layers from the top and the bottom with a slip ring placed on top of the frame (Fig 11). The slip ring has 6 wires, which is enough to control the LEDs as Adafruit Neopixels only require 3 wires – GND, 5V, DATA IN. The strips of LEDs were soldered in series in order to maintain the ability to address the LEDs as an array (Chapter 3.2.3 Equation 1). An additional wire (DATA OUT) was used to receive data from the Hall effect sensor which was planted at the bottom layer, whereas the GND and 5V wires of the sensor were connected to the same slip ring wires as the GND and 5V of the Neopixels respectively.

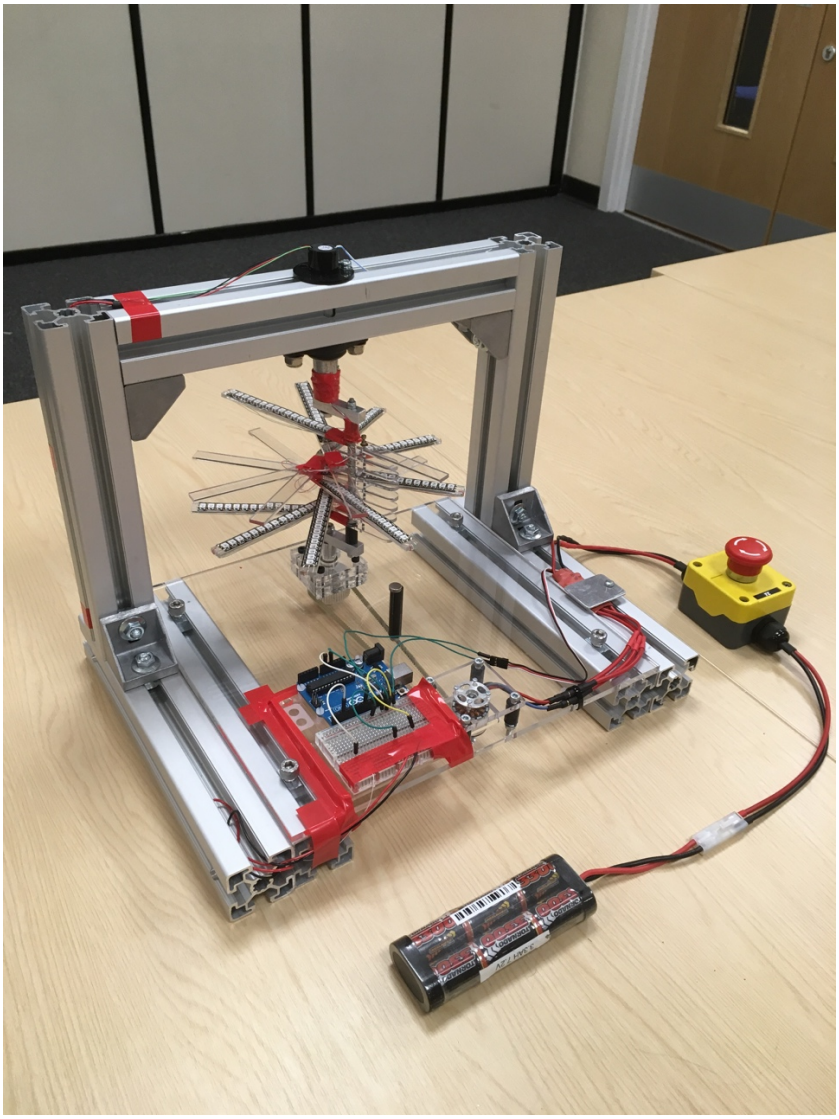

Fig. 11. Image of the whole mechanical design consisting of a metal frame and layers assembled together. The slip ring is mounted through the top part of the metal frame

Two pulleys (the driver pulley and the driven pulley) were placed at the bottom of the frame under the layers, this way allowing the DC motor spin the layers from a distance (Fig 12). The layers are mounted on top of the driven pulley so that the motor does not have direct contact with the layer structure. This mechanism reduced the amount of weight, pressure and load provided by the layer structure that would negatively affect and slow down the motor.

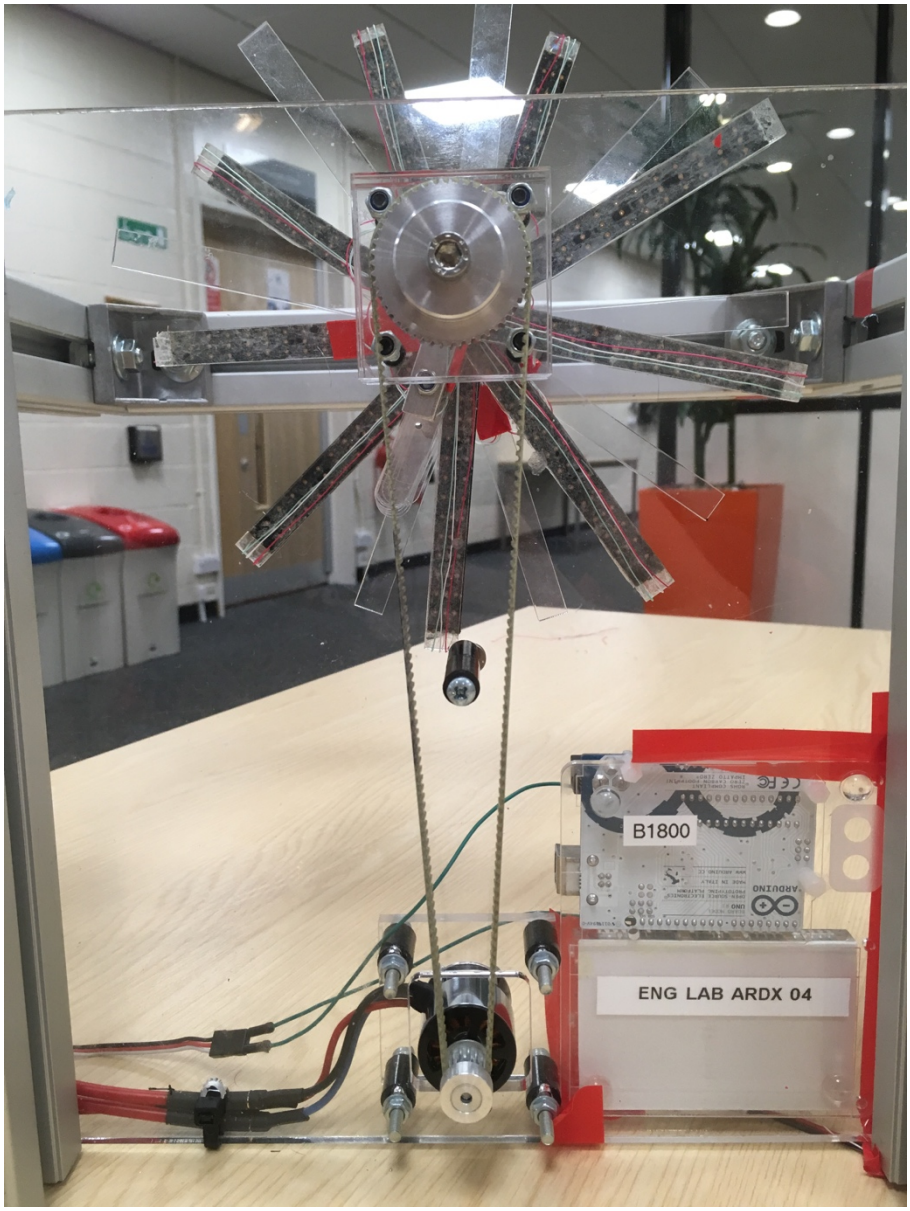

Fig. 12. Image of the pulleys joined together by a belt. The driver pulley (at the bottom) is spun by the motor which automatically spins the driven pulley (at the top) with the layers mounted on top of it

4.2 Hardware design

4.2.1 Electronic parts

The motor that was used in the design was the A2212/13T brushless DC motor [23] with an Overlander 3300mah 7.2v [24] battery pack powering it. The main reason of choosing this DC motor is the good 1000 RPM/V ratio. The second reason was that its efficiency is almost the same when using stronger batteries [23]. In addition, the DC motor shaft provided good adaptability with the small pulley that was used. The main reason of choosing this battery pack was that it provides the voltage that is required by the DC motor as a minimum (7.2v [24]). The battery is rechargeable so this also provides reusability. The DC motor was connected to the ESC (electronic speed control) circuit that varies the speed and the direction of the electric motor. In Arduino's hardware code, the ESC also can be controlled by a servo library which provides a higher level abstraction and simplifies the programming process.

4.2.2 Circuitry

As discussed in Chapter 4.5.2, there are 4 wires connecting the microcontroller to the spinning part of the design: GND, 5V, DATA IN (LEDs), DATA OUT (Hall effect sensor). In addition to that, two more wires need to be connected between the Arduino board and the ESC of the motor – GND and DATA IN. Overall, the final circuit diagram is represented in Fig 13.

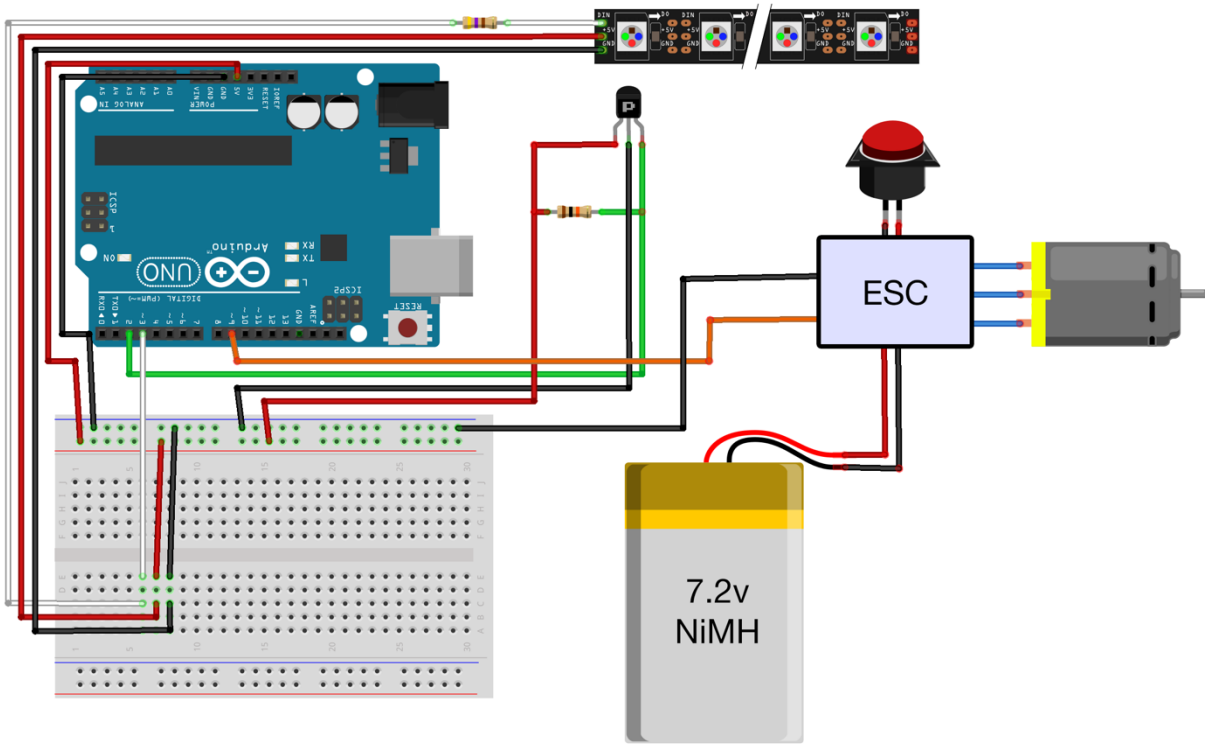

Fig. 13. Complete diagram of the circuitry of the system. The 9 layers of LEDs connected in series are represented as a single layer (top of the diagram)

The Digital Pin 2 is connected to the Hall effect sensor, the Digital Pin 3 is connected to the NeoPixels, the Digital Pin 9 is connected to the ESC of the motor. 5V and GND Pins are connected to the top rows of the breadboard in order to allow more wires to connect to either 5V or GND pins. There is a 10KΩ resistor between the DATA OUT and 5V wires of the Hall effect sensor. Similarly, there is a 470Ω resistor between the DATA IN wire and the Digital Pin 3 of the Arduino Uno. In order to save space in the diagram, all the layers were represented as one, because all the layers are connected in series and there is no change within the circuitry (Fig 13).

4.3 Software design

4.3.1 Agile life-cycle

In order to develop steadily and efficiently, a software development methodology must be chosen. The two most common application development life-cycle models are – waterfall and agile. At this point, we assume that the reader is already familiar with both of the models. The requirements of this project have already proven to be changing at a rapid rate and,

according to one of the goals of this project – system must be ready to be deployed (Chapter 1.3). Thus, it was chosen to develop by using the agile development technique. The software must be adaptable with any new changes that might be done and must be easily editable without affecting other functionalities of the system. To minimise the likelihood of potentially losing source code and to keep track of the performance, a source tracking system GitHub was used.

4.3.2 Three.js vs Other

As discussed in Chapter 3.3.2, there are several options to choose from for a light-weight 3D graphics library for web development. So far, the most popular ones are the Three.js and Babylon.js libraries. The Three.js library provides an easier code framework. In addition, "Babylon.js subtly differentiates itself by focusing on traditional game engine requirements like engines and custom lighting" [25]. Based on these two factors, the Three.js engine was chosen to be used in this project.

As mentioned in the previous chapter, the total number of pixels is 6480 (Chapter 3.1.4). Therefore, the decision was to draw 6480 invisible cubes everywhere inside the cylindrical space, so if a new shape is created, the engine would check for collisions between the new shape and the invisible cubes. After a user finishes his design, the system sends data (ids and colours) about the invisible cubes that were in collision with the shapes that were drawn. The coordinate ids are given in a particular predetermined order (Fig 14).

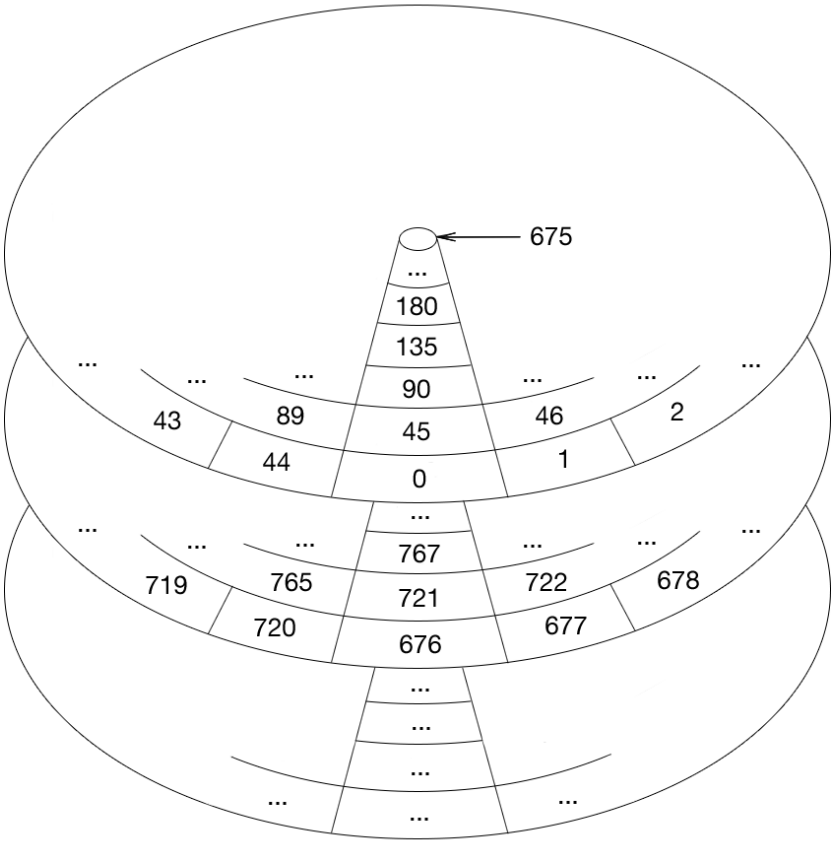

Fig. 14. Scheme representing how ids of the coordinates are given starting from the top layer

There are 5 pixels (coordinates) between each layer, and there are 9 layers, so the total number of coordinates that a single LED can represent is 45. Therefore, the outermost LED

on the first layer has an id 0, then the one next to it – id 1, etc. (Fig 14).

4.3.3 Front-end and back-end

The development of the client-side application consisted of developing the UI, and the 3D graphics side of it. The 3D graphics part of the application was purely built in Three.js. To develop the base for the website application, HTML was used. To build the UI (user interface) part of it, the JavaScript programming language and CSS were used. To include sliders that allow users to change values within shapes – jQuery library was used. The application is web-based, therefore, an Apache server was used that supports PHP (server-side programming language). In order to store data of users' projects and shapes – mySQL query language was used with the help of phpMyAdmin that is a graphical administration system for the database. To send data from the web application to the Arduino board via Arduino's serial port, phpSerial library was used, that, after a 2 seconds delay, would send data through.

The application consists of 3 parts:

1. the login screen, in which users able to register or login
2. the dashboard, in which all user's projects are listed
3. the playground, in which users are able to draw 3D designs

4.4 Changes

4.4.1 Abstract changes

There were some major changes that were made to the project and its architecture. After testing the Adafruit Neopixel strip LEDs, it was apparent that the refresh rate (turning the LEDs on and then off again) was too low in order to maintain the correct drawing of images – the pixels appeared to be too long. Later, the "Adafruit DotStar Digital LED Strip - Black 144 LED/m - One Meter - BLACK" [26] consisting of 144 LEDs and a 20KHz refresh rate (which is 500 times faster than the NeoPixel refresh rate) was used. However, the pixels drawn were only slightly smaller than the ones drawn by the NeoPixels. After some more testing and developing, the *FastLED.h* [27] library was used which was specifically made for Adafruit products. This time the lengths of the pixels were much smaller, however, the pixels were still too long. Finally, several normal LEDs were mounted on top of the layer structure in order to test if this sort of design would ever work as a proof-of-concept solution. Although, the normal LEDs worked correctly by drawing pixels with correct lengths, Arduino Uno does not provide enough output pins and the whole design (when using 144 LEDs) would require too much wiring even using the Charlieplexing technique.

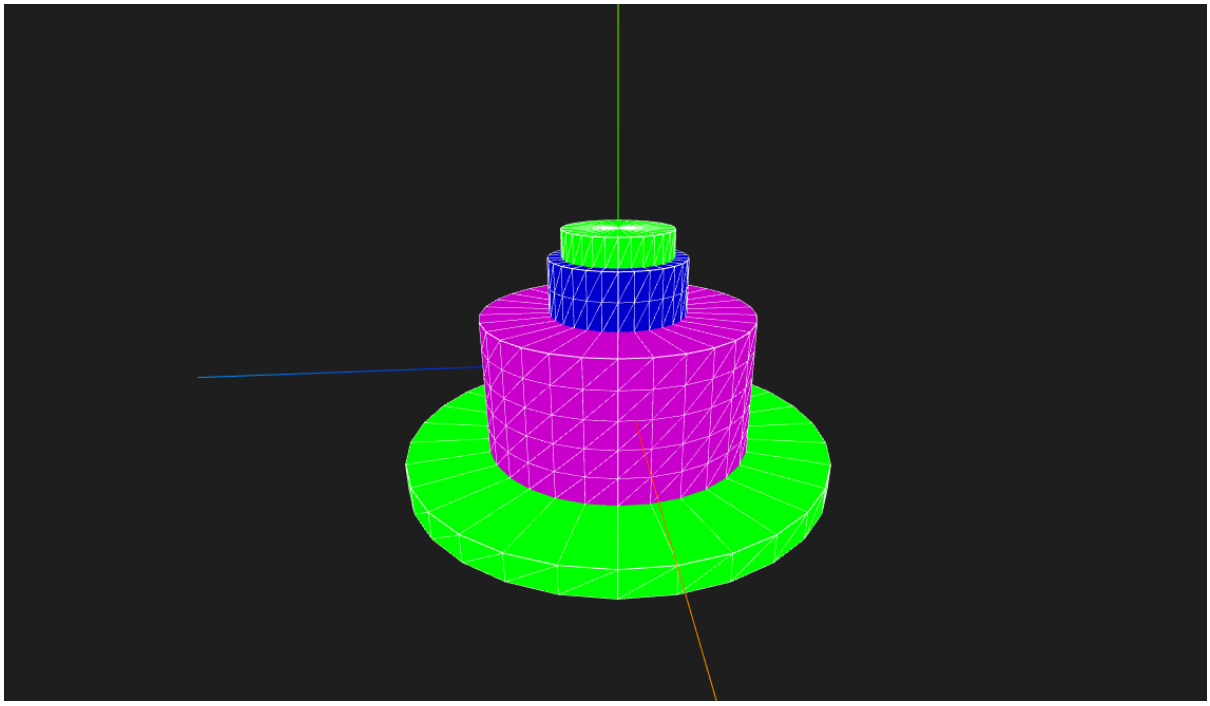

Fig. 15. An example of a 3D shape design after the requirements of the software application have been changed

Finally, it was decided to use the NeoPixel strips but to change the design, so a user defines the widths of each layer (starting from the pixel in the center), and the colours of them (Fig 15). That way, the LEDs work statically – the required LEDs are turned on and there are no further changes.

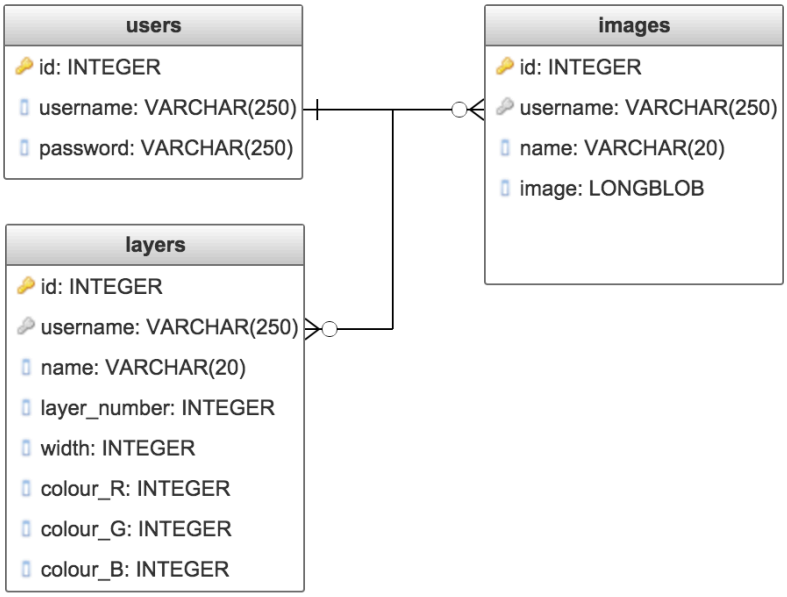

Fig. 16. Database schema of the database

According to the new requirements of the system, the final database schema consists of 3 tables: *users*, *layers*, *images* (Fig 16). The *users* table stores usernames and encrypted (one-way hashed) passwords of users. In order to maintain confidentiality, each password typed in is hashed and compared to the one that is in the database, and if they match – user is authenticated. The *layers* table stores data about each layer. The *images* table stores data about small images that are shown on the dashboard which are generated after a project is created or updated.

4.4.2 Hardware changes

On the hardware side, the Hall effect sensor was removed from the design, so there was no need to track the RPMs anymore because of the new requirements (the LEDs are statically turned on).

4.4.3 Software application changes

In order to comply with the new requirements of the system, the 3D graphics side of the client-side application was changed. Now, users were able to create 3D shapes that consisted of 3D cylinders (Fig 15) placed on top of each other, with the top cylinder representing the top layer. The default width for each layer was set to be 10 and the colour – pink.

4.4.4 Communication protocol

The communication protocol to send data (with encryption) from the software application to the Arduino was as follows:

total_no_of_layers, layer_1_width, colour_R, colour_G, colour_B,
layer_2_width, colour_R, colour_G, colour_B,
layer_3_width, colour_R, colour_G, colour_B,
layer_4_width, colour_R, colour_G, colour_B,
layer_5_width, colour_R, colour_G, colour_B,
layer_6_width, colour_R, colour_G, colour_B,
layer_7_width, colour_R, colour_G, colour_B,
layer_8_width, colour_R, colour_G, colour_B,
layer_9_width, colour_R, colour_G, colour_B

This data is decrypted on the Arduino before updating the LEDs. After the data has been decrypted, an array of *structs* is created, which is correctly interpreted and the LEDs are updated. We shall now move on to the evaluation part of the project.

Chapter 5 – Evaluation

5.1 Requirements

In order to correctly evaluate the project, we need to see what requirements were met. We will firstly discuss the functional requirements.

The first part of the project was designing and constructing the mechanical part of a semi-hologram device. The first, second and the fourth functional requirements of the mechanical design (Chapter 3.1.1) were met by this project. The system works in any light conditions because of the NeoPixels used, there is an ability draw shapes anywhere inside the cylindrical space because of the metal arm holding the layers from the side, and finally, microcontroller does not spin because of the slip rings that were used. The third requirement was not met because of the light metal material used to build the frame, so there still are high vibrations caused by the motor. The first and the third non-functional requirements were met. The whole construction consists of cheap parts (costing no more than £100 total, including the 2 packs of new LEDs), the device is reusable because of the strong plastic layers, the metal pulleys and the frame that were used. The device is not portable and easy to carry (weighs around 5kg), so the second requirement was not met.

The second part of the project was designing and connecting the hardware components. All of the functional requirements of the hardware design were met. The Arduino board was able to keep track of RPMs and send the precisely timed signals to the LEDs because of the Hall effect sensor that was used, is able to control the motor because of the ESC used, is able to receive data from the client-side application because of the phpSerial library used, and finally, the motor signals do not interfere or affect the LED signals because of the different pins used. The second, third and the fourth of the non-functional requirements were met. The hardware design supports open source development because of the open source Arduino and Neopixel libraries used, system is reliable because of the premade microcontroller board used, design provides good developer usability because of the many comments within the code and the small amount of code overall. The first requirement was no met because of the lack of testing on the hardware side (e.g. test that the encrypted data is not too long, the layer ids are in bounds).

The third part of the project was designing and developing the software application. All of the functional requirements were met. There is support for three-dimensional objects, users are able to create/update/delete their projects, rotate shapes and change their colours because of the Three.js library that was used. Users are able to login so different users' projects do not clash with each other because of the database used and the login mechanism that was built in. All of the non-functional requirements were met. An application is lightweight and easy to use because of the simplified UI that was developed, is relatively secure because of the hashing technique of the passwords of users, provides good testability because of the exhaustive unit and automation tests that were written, provides good scalability because of the Apache server and the mySQL database used which have a good level of reliability, and

finally, provides good maintainability because of the data being stored on a database, rather than locally.

5.2 Software testing

5.2.1 Unit tests

The software application has both unit and automation tests. The unit tests were relatively exhaustive tests, so they were written for the most possible scenarios (excluding the rare ones) with a qUnitJS library [28]. It mostly tests the functionality of the client-side, so the behavior of the most important functions written in JavaScript were tested.

5.2.2 Automation tests

Similarly, the automation tests were also relatively exhaustive tests, that focused on most of the possible scenarios when navigating through the web application. Thus, the automation tests tested the application purely from a user's perspective. Selenium library [29] in addition to Gecko/Chrome drivers [30] were used in order to run the simulations. The tests mainly tested the happy case scenarios and focused on testing whether the pixels generated comply with the protocol, so there was no testing done on the rare cases.

5.2.3 Likely sources of error

There still are many likely sources of error on the software application, which might negatively affect the system or even break it. There still is a high chance of braking or taking over the system with not much effort or knowledge required. The system is vulnerable to potential DoS attacks or SQL injections, that might be used to crack the application. However, security was not the priority of this project, therefore, there was not much emphasis on it.

5.3 Hardware testing

5.3.1 Overall tests

On the hardware side, there were no tests written, because we assumed at the start of this project that users have no ability to interact with the hardware side of the device. The potential tests that could have been written are – testing whether the encoded data that was sent was actually received, the data complies with the protocol and the data that was received has no errors in it. We also assume that data that is coming from the software application has already been tested.

5.3.2 Likely sources of error

Although, hackers might still be able to get access to the code and change it. Despite our assumptions, there still might be a middle-man that would change data in between the client-side application and the microcontroller board. This way he would get access to data from all the projects on the server. Again, although there is a high chance of attackers potentially breaking the system, device security was not the priority of this project.

Chapter 6 – Reflection and Conclusion

6.1 Management

6.1.1 Milestones

Overall, the management side of the project can be considered to have been done well. One important observation is that this project focused more on the research part of a new technology so there were no clear guidelines on how to approach this problem. Research projects usually have rapidly changing requirements and the requirements of this particular project have changed many times during the process. Therefore, the agile approach was a good decision which saved time afterwards when the conditions kept changing. Agile life-cycle allowed the project to be adaptable to new changes which increased the efficiency and regular pace of development. As discussed in the previous chapter (Chapter 5.4), most of the functional and non-functional requirements were met with 17/25 ratio which is 68% and could be considered a good fulfillment of the requirements. However, this ratio represents the fulfillment of the initial requirements which might not be correct in order to achieve what was intended. So, the level of meeting all the requirements varied during the process, with some of the requirements being fulfilled at first, but then being discarded when the sub-goals of the project changed.

6.1.2 Goal fulfillment

However, the main goals (Chapter 1.3) of this project have not changed and were all reached. The first goal was reached because there was a display built that is able to draw 3D shapes, especially, in the center of the cylindrical space of the system. The second goal was also reached because of the personal gains in knowledge about web development and exhaustive testing. In addition, the main issues that were described during the development of this project were – the refresh rate of the LEDs and instability of the system (vibrations). The third goal was met because of the client-side application that was developed during the process. The fourth goal was reached because of the project being nearly ready for further deployment and both the mechanical design and the software application being completely finished. The last, fifth, goal was reached because of the evaluation of the results and an outline for further research.

6.2 Knowledge

During the development of this project, I personally improved both my viewpoint to the problem and the technical skills. I learned that during the early phase of the development, there are no clear requirements and that the requirements might change quickly, especially, in research-type projects. Another valuable lesson I learnt was that the requirements are the cheapest asset to fix, so the earlier the correct requirements are indicated, the more time and money is saved. On the technical side, most of the skills that I have gained were the microcontroller-level development knowledge and website (front-end and back-end) design.

6.3 Conclusion

In summary, this project was challenging because of the lack of knowledge I had before

starting the work. One of the most challenging tasks was to come up with an abstraction that converts an imaginary 3D image into a real life representation of it. Most of the initial requirements were met after finishing the work on this project and all the goals of this project were reached. Ultimately, based on the reasons stated in this chapter, the project was a success.

References

[1] voLumen - volumetric 3D display device, accessed: 2017-04-28. Url: https://hackaday.io/project/1737-volumen-volumetric-3d-display-device
[2] 3D POV display ... work in progress, accessed: 2017-04-28. Url: https://www.youtube.com/watch?v=rlptJzWwCXg
[3] Development of a 3D Display, accessed: 2017-04-28. Url: https://github.com/mbjd/english-paper/blob/master/paper.pdf
[4] Persistence of Vision - Computer Monitors, accessed: 2017-04-28. Url: https://en.wikipedia.org/wiki/Persistence_of_vision#Computer_monitors
[5] S. Ost, Gif Shop is Youtube For Everyday Animators, accessed: 2017-04-28. Url: https://www.fastcompany.com/1775798/gif-shop-youtube-everyday-animators
[6] Filmstrip, accessed: 2017-04-28. Url: https://en.wikipedia.org/wiki/Filmstrip
[7] P. Read, M. P. Meyer, Restoration of Motion Picture Film, accessed: 2017-04-28. Url: https://books.google.co.uk/books?id=jzbUUL0xJAEC&pg=PA24&lpg=PA24&focus=viewport#v=onepage&q=visual%20system%20can%20process&f=false
[8] M. Smugowska, Holovect – DIY Laser-Based Holographic Display, accessed: 2017-04-28. Url: https://www.startingthingsup.com/en-gb/holovect-diy-laser-based-holographic-display/
[9] Dedi 3D Holographic Advertising/Transparent Screen for Projector/Hologram Glass Window Display, accessed: 2017-04-28. Url: http://dedi2013.en.made-in-china.com/productimage/pByxnUYrFJRH-2f1j00tZPEclkhuzpT/China-Dedi-3D-Holographic-Advertising-Transparent-Screen-for-Projector-Hologram-Glass-Window-Display.html
[10] B. Dettling, 3D POV Display, accessed: 2017-04-28. Url: https://www.youtube.com/watch?v=bCETWNgBxbI
[11] M. Mali, viSio – volumetric 3D display device, accessed: 2017-04-29. Url: http://maxmali.com/work/
[12] L. Calderone, The Potential of Holographic Displays, accessed: 2017-04-29. Url: http://www.hometoys.com/article/2014/02/the-potential-of-holographic-displays/2123/
[13] Adafruit NeoPixel Digital RGB LED Strip 144 LED - 1m Black - BLACK, accessed: 2017-04-29. Url: https://www.adafruit.com/product/1506
[14] Arduino UNO & Genuino UNO, accessed: 2017-04-29. Url: https://www.arduino.cc/en/main/arduinoBoardUno
[15] P. Burgess, Adafruit NeoPixel Überguide, accessed: 2017-04-29. Url: https://learn.adafruit.com/adafruit-neopixel-uberguide/arduino-library
[16] Scalability - Horizontal and vertical scaling, accessed: 2017-04-29. Url: https://en.wikipedia.org/wiki/Scalability#Horizontal_and_vertical_scaling
[17] Unity 5.6, accessed: 2017-04-29. Url: https://unity3d.com/
[18] Desktop/Android/BlackBerry/iOS/HTML5 Java game development framework, accessed: 2017-04-29. Url: https://libgdx.badlogicgames.com/
[19] jMonkeyEngine - A cross-platform game engine for adventurous Java developers, accessed: 2017-04-29. Url: http://jmonkeyengine.org/
[20] Java 3D Parent Project - Project Kenai, accessed: 2017-04-29. Url: https://java3d.java.net/
[21] three.js - Javascript 3D library, accessed: 2017-04-30. Url: https://threejs.org/
[22] BabylonJS - 3D Engine based on WebGL/Web Audio and Javascript, accessed: 2017-04-30. Url: https://www.babylonjs.com/

[23] 400 Size Brushless Outrunner Motor A2212/13T Technical Data, accessed: 2017-04-30. Url: http://www.batteryheatedclothing.com/pages/a2212-13t-technical-data.html
[24] Nimh Battery Pack SubC 3300mah 7.2v Premium Sport, accessed: 2017-04-30. Url: https://www.overlander.co.uk/nimh-battery-pack-subc-3300mah-7-2v-premium-sport-now-with-deans-connector.html
[25] Three.js and Babylon.js: a Comparison of WebGL Frameworks, accessed: 2017-04-30. Url: https://www.sitepoint.com/three-js-babylon-js-comparison-webgl-frameworks/
[26] Adafruit DotStar Digital LED Strip - Black 144 LED/m - One Meter - BLACK, accessed: 2017-04-30. Url: https://www.adafruit.com/product/2241
[27] Fast, easy LED library for Arduino, accessed: 2017-04-30. Url: http://fastled.io/
[28] QUnit: A JavaScript Unit Testing framework., accessed: 2017-04-30. Url: https://qunitjs.com/
[29] Selenium WebDriver, accessed: 2017-04-30. Url: http://www.seleniumhq.org/projects/webdriver/
[30] ChromeDriver - WebDriver for Chrome, accessed: 2017-04-30. Url: https://sites.google.com/a/chromium.org/chromedriver/