Case Study Report: Drawing Robot

Group 13

Jijing Hon, Hugo Millet, Clément Noblet, Pranjal Samant, James Whitehouse

B50RO & B51RO Robotic Mechanical Systems

Due date: 25/04/2023

Abstract

The aim of this case study report is to study and design a robot required to draw a portrait on the rounded surface of a cylinder. The system design was inspired by several existing printers such as a dot matrix printer, which prints using dots equivalent to pixels of a portrait. It was decided to have the cylinder laid horizontally across two walls to reduce the tension for the stepper motor. The design eliminates unnecessary vibrations causing further inaccuracy. The designed system pixelates the given portrait or image and transfers the codes to the actuator in 'pen down' and 'pen up' signals like a dot matrix printer. Although the results were not completely identical to the original portrait, the overall result has the main characteristics of the original portrait.

Table of Contents

Lists of Figures

1 Introduction

The robot being built is a model of a dot matrix printer (DMP). The model being produced by the group has been modified so that it can print images on cylindrical surfaces. This means that a different degree of freedom must be added to the design of a DMP. Normal DMP's use the x and y-axis for movement around the page, and the z-axis for the pen or ink movement up and down. The new model will move along the y-axis normally, and then a motor will spin the cylinder around the y-axis. The pen movement in the z-axis will be the same as a traditional DMP.

2 Objective

The objective of the Case Study was to design and create a robot that was within the budget of £60 while using the fundamentals learned in the course. In the case of the drawing robot that was created, this was achieved using CAD and MATLAB, along with principles like Kinematics. This report demonstrates the uses of each of these topics and more that were used to create the robot described.

3 Literature Review

Dot Matrix Printers started becoming more popular in the 1980's and are a simpler form of the printer used today (Allen 1987). In Forensic Science International, Volume 35, Issue 4, M.J. Allen states that Dot-matrix printers are becoming widely available (Allen 1987). These DMP's used steel pins in a straight line (Allen 1987). These pins were then able to be controlled individually to create symbols and letters by striking an inked ribbon (Allen 1987). The same concept is used for the model being created. The pin will instead be a marker which is controlled by signals from the controller. Then the marker will be moved up and down on the y-axis to then create images on the cylinder. The cylinder will be rotated so that the marker will be able to reach anywhere on the cylinders face. An old style DMP uses a "character matrix" in which the pins are in a matrix of 9x7 dots as seen in the image below (Allen 1987).

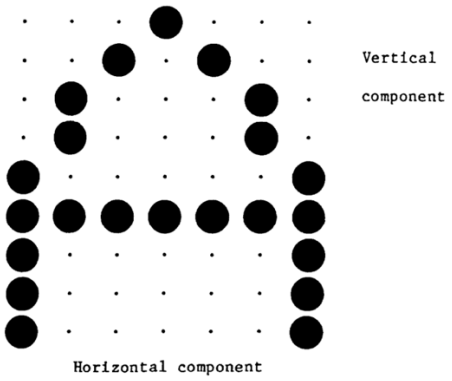

Figure 1: Example of 9x7 character-matrix showing letter A [1]

This will be the basic concept that the drawing robot uses. The cylinder will be mapped with a matrix which will be populated by dots to create an image by the drawing robot. The drawing robot will be able to draw in single colors at a time. This means that if there is a black outline, then the robot will draw that

first and then fill in the rest of the image. If color is needed, the marker will need to be switched out and the matrix will need to be updated for the certain color.

Another technique is explained by White et al. in Behavior Research Methods, Instruments, & Computers. They portray a technique used for gray-scale graphics using a DMP (White et al. 1984). White et al. used different sets of printing symbols, taking the reflectance of each of the symbols, and then using a photometer to see the lightness or darkness scale of the symbols when plotted in the matrix at different densities instead in intensities (White et al. 1984). These symbols are different for each level and are described in the table below (White et al. 1984).

A Set of Printing Symbols for a Seven-Level Gray Scale

Level	Printing Symbol	Description	Reflectance	Log Reflectance
7	@	"at"	.42	−.381
6	%	per cent	.48	−.318
5	J	capital J	.56	−.251
4	>	greater than	.62	−.211
3	"	quotation mark	.70	−.152
2	'	apostrophe	.79	−.101
1		space	.89	−.051

Figure 2: Different symbols for the 7 levels [2]

The table shows what will be portrayed Figure 3. The different symbols have different levels of reflectance, which is then used to create a higher density dot matrix, thus creating different levels of grayscale (White et al. 1984). These then create density maps like the one shown below (White et al. 1984).

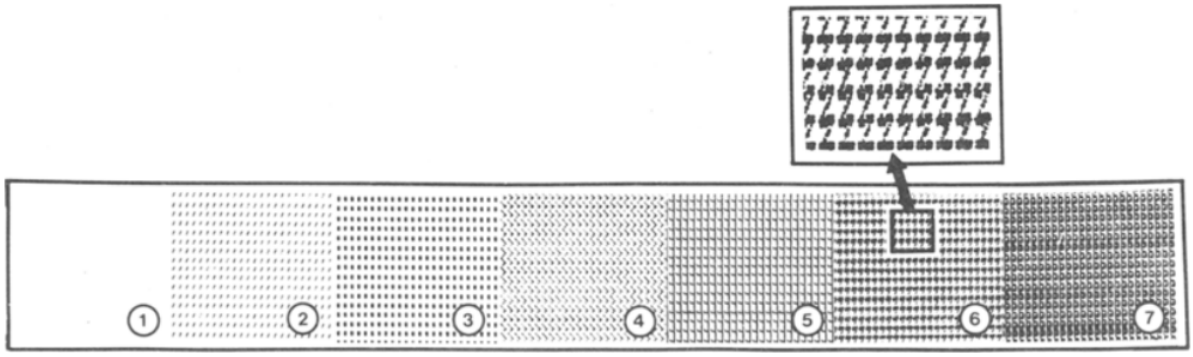

Figure 3: 7-level Gray scale on DMP [2]

This is theory is used in the drawing robot, but without symbols. The drawing robot will cluster points together to make certain points of the picture being created darker. The robot will do the opposite to make the picture lighter in certain areas as well.

Sundheim writes about graphics while using a DMP in the Journal of Chemical Education Volume 61, Issue 6, 1984. He states that high quality graphs, mathematical and chemical symbols for equations etc. generally required special, high-cost printers in 1984, but with the new wave of DMP's, the costs for printing these visuals were dramatically reduced (Sundheim 1984). It is stated that on most of the DMP's there is a function that allows the pins to be controller by the microcomputer directly which allows for the high detail required for said graphs and symbols (Sundheim 1984). This is like the drawing robot being created as it will use info directly sent to the microcomputer, in this case, an Arduino, and will then map and printer then image clearly on the cylindrical object. The difference is the image being drawn by the drawing robot will be constrained by the marker size being used to show the fine details of the image. If a smaller sized marker was being used, then a higher detail could be achieved.

4 Design Process

4.1 Initial designs

One of the challenges to drawing on the cylinder is the curvature on the surface. From researching existing printing systems there were a few inspirations obtained. Below are the initial designs that were discussed during the design process.

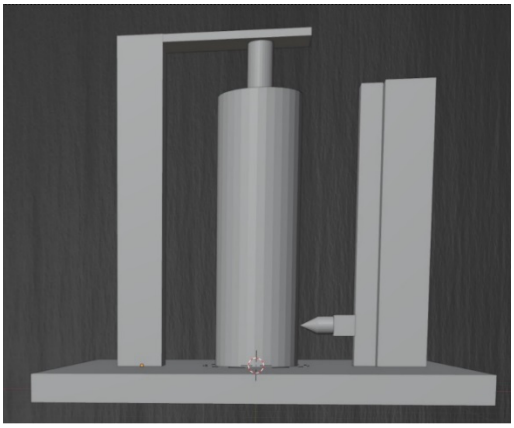

Figure 4: Design 1

Design 1 (Figure 4) consists of a shaft through the cylinder portrait, and a moving pen system separate from the cylinder holder.

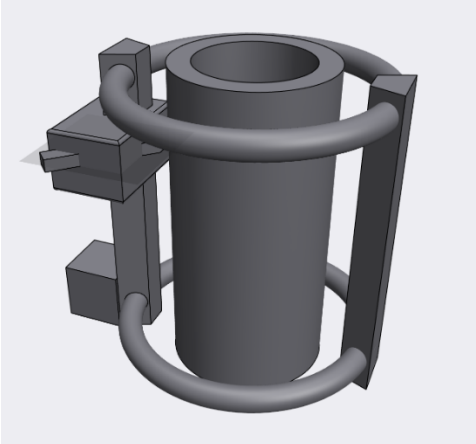

Figure 5: Design 2

Design 2 (Figure 5) comprises 2 circular rings around the cylinder, and 2 poles to hold up the 2 rings. This provides structural support to the drawing component.

Both designs have 3 degrees of freedom, where the drawing mechanism acts along the Z-axis, the movement of the pen along the X-axis and the cylinder rotation about the Z-axis. The system requires the cylinder to be fixed onto a base. When transporting a motor on the Z-axis, the component can be moving along or against gravity. Hence, the weight of the components must also be accounted for the torque. The weight of the pen, the holder and the motor must not total and exceed the ability of the chosen motor.

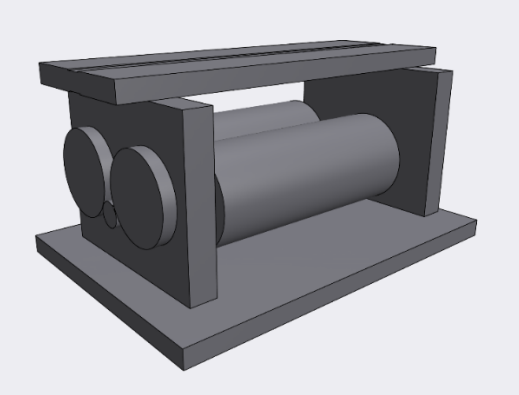

Figure 6: Design 3

Design 3 (Figure 6) was an inspiration behind bottle printers. The 2 cylinders in the design is used to rotate the bottle in segments, whilst the top piece holds the drawing system. This design wasn't chosen as the handling of the cylinder may be inaccurate. In other words, the drawing pen may drag the cylinder, causing the cartesian location of the cylinder to change, hence resulting in a shifted drawing.

5 Final Design Description

The final design and the exploded drawing of the robot is presented in Figure 7 and Figure 8 respectively. The design consists of several parts available off-the-shelf e.g., synchronous belt, rods, and ball bearings.

A few specific dimension components, such as the synchronous pulleys, pen holder and cuboid stands were 3D printed using PLA filaments.

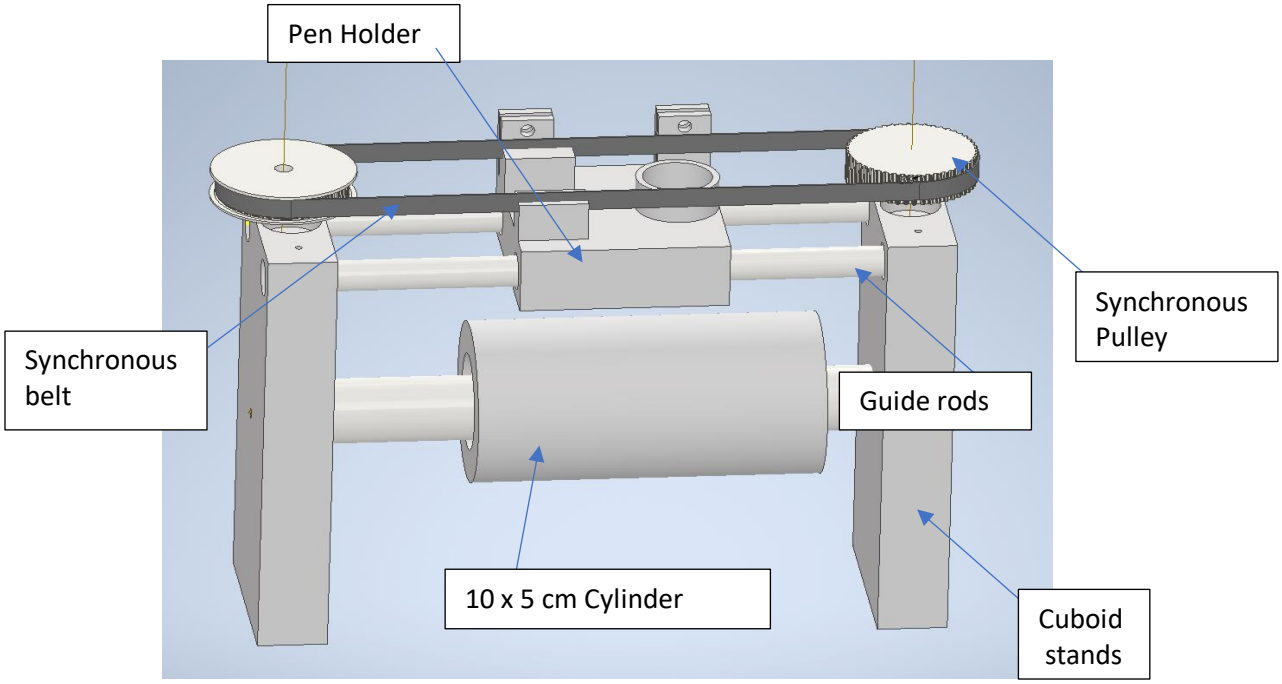

Figure 7: 3D CAD labelled view of design.

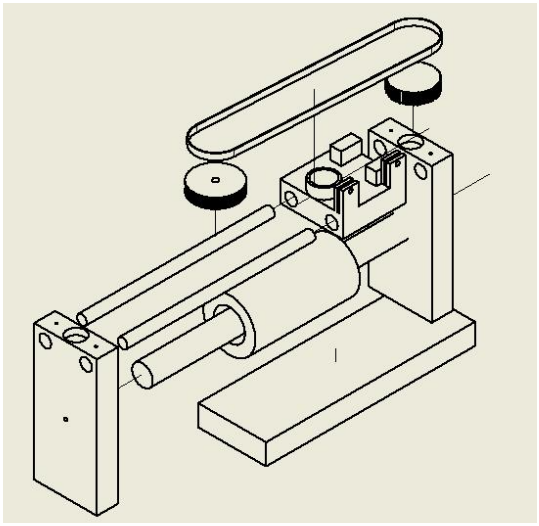

Figure 8: Exploded CAD drawing.

5.1 Kinematic Modelling

The mechanical modelling was done without considering the system allowing to lower and raise the pen. This choice has been made because in our case the position of the pen holder and the cylinder are the important things for the positioning of the writing.

At the level of the connections, parts 1 and 2 are mechanically connected by two cylindrical connections, which is equivalent to a prismatic connection. Two cylindrical links allow us a better rigidity of the system, a better precision but also the possibility to pass the pen to write on the cylinder.

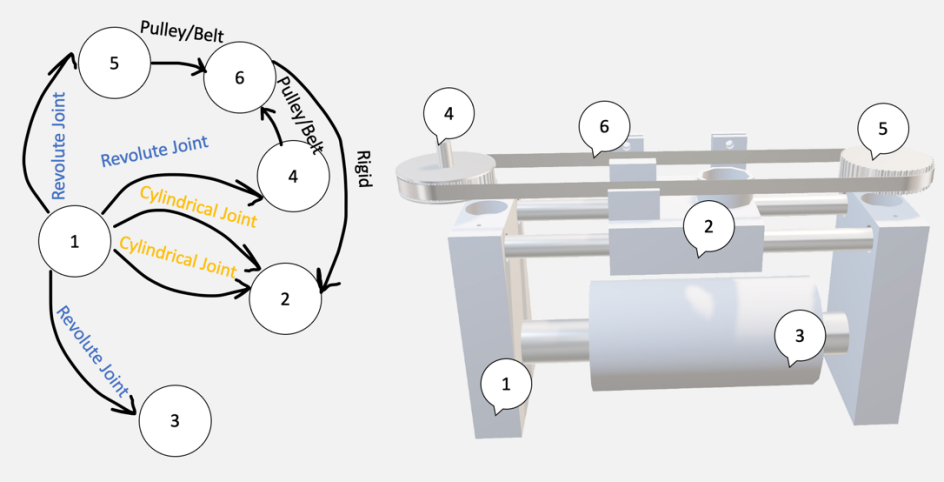

Figure 9: Diagram of mechanical links

.

The system can be summarized by 2 mathematical relations describing for one the movement of the pen holder according to the variation of angle of the stepper motor, and the other the rotation of the cylinder according to the servomotor.

$x = 180 \times \theta_1 \times d$ (Equ. 1)

$\theta_{Cylinder} = \theta_{Servo.}$ (Equ. 2)

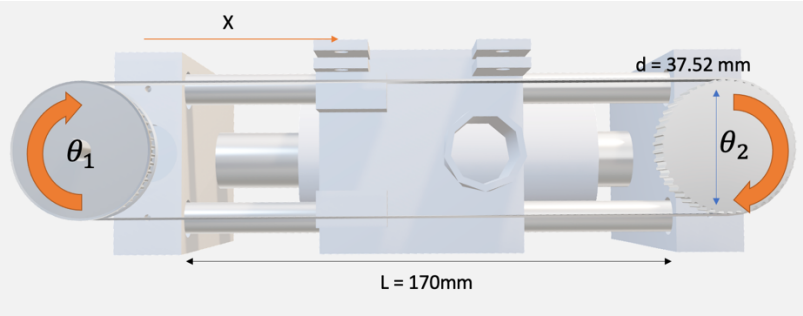

Figure 10: Diagram of the pen holder movement

By considering the first equation, the number of revolutions that the motor must make so that the pen holder moves on the whole printing range was found as seen below in Equation 3.

$N_{max} = \frac{\pi d}{L} = 0.70\text{rev}$ (Equ. 3)

5.2 Robot Body

In Figure 11, the cylinder was held up by two 3D printed blocks. Each of these blocks had holes in the top to allow for the supports which held the pen holder. The white block seen below had extra holes. One in the top and one in the middle to allow for bearings to be attached. This allowed for the gears to rotate on top for the pen movement, and for the cylinder to rotate. The motors were then attached using zip ties, while the Arduino was attached to another 3D printed block. Detailed CAD drawing for the different parts of the robot can be found in the Appendix.

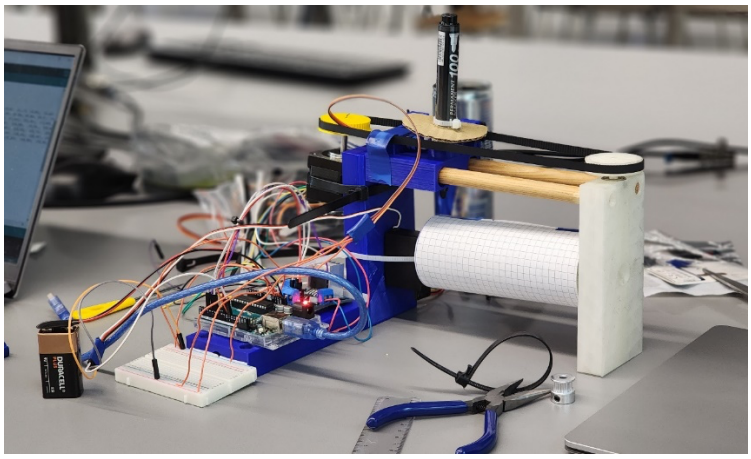

Figure 11: Final design in working

5.3 Pen holder

The pen holder is a complex design that is required to fit several joining parts which includes, the belt, the pen, the servo motor and the 2 guide rods. Figure 12 shows the pen holder with the different components fitted with the pen holder.

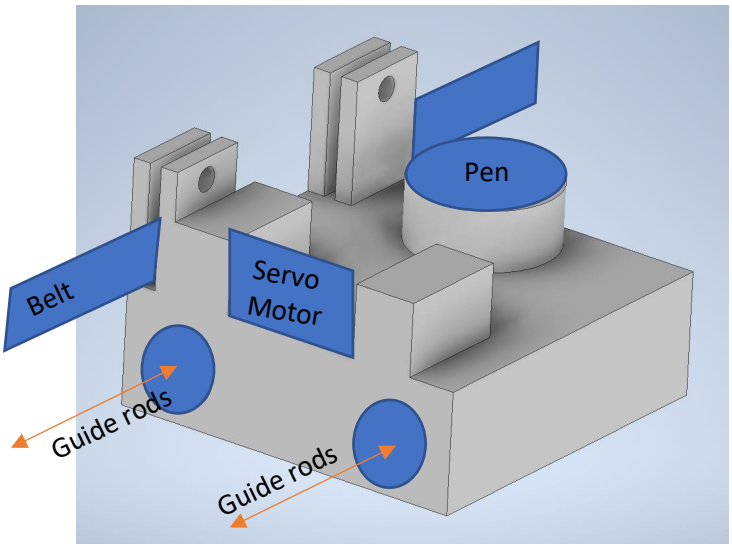

Figure 12: CAD view of pen holder

5.4 Cost analysis

Components	Source/ Cost
Marker Pen	£2.00
Ball Bearing	£3.60
Guide rods	Heriot-Watt University
Servo motor	Heriot-Watt University
Stepper motor	£22.68
Synchronous belt	Heriot-Watt University
Arduino – breadboard	Heriot-Watt University
3D Printing	Heriot-Watt University
Power supply	£8.15
Total Cost	£34.43

Table 1: Costings for Robot Design

As seen in Table 1, the cost to produce this robot is relatively small. While half the components were supplied, so it came at zero cost. The cost for these products would be reduced further if bought at a wholesale price. With the total cost of £34.43, it is expected for the number to reduce by a certain percentage depending on the materials available. The biggest cost of the robot was the stepper motor. In the future, a new motor could be researched and then implemented to reduce the total cost of this robot. The total cost was under the budget given of £60, but if the materials provided by the University had to be bought, the total cost of the robot would have been significantly closer to this value.

6 Electrical Design

6.1 Introduction

Any project's electrical design is critical since it serves as the control system for all electronic components. In the Case Study, the proper operation of the prototype is significantly reliant on the perfect coordination of these components, ensuring a smooth execution of the procedure. This was accomplished by combining basic electronic components and a microcontroller, allowing for demonstration of the project's practical applications in a realistic manner.

6.2 Components

1. **Arduino Uno R3:** It is a microcontroller that provides a framework on which the user can create electronic projects. It is used in this project to operate the stepper and servo motors.
2. **L298N Motor Driver:** To control motors and stepper motors, a dual H-bridge motor driver integrated circuit (IC) is used. By sending PWM signals to the input pins of the motor, the user may regulate its direction and speed. It features a total of 4 input pins, including 2 for direction and 2 for speed. It was utilised to manage a hybrid stepper motor.
3. **Hybrid Stepper Motor (9.6V):** The greatest qualities of both permanent magnet (PM) and variable reluctance (VR) stepper motors are combined to create a hybrid stepper motor. High precision, accuracy, and torque are all strengths of theirs. In this project, it is being used to move synchronise pulley and belt mechanism.
4. **Micro-Servo Motor:** This motor is capable of precise movements and are designed to rotate between 0 and 180°. The main advantage of using a micro servo motor is its small size and weight. That is why it is used to control the pen mechanism of the robot.
5. **RC Servo Motor:** These motors have good precision control and positioning capabilities. It is used to rotate the cylinder on which the drawing sheet is attached.
6. **Power Supply:** A power supply of 9V was used in this project to power the L298N motor driver which controls the Hybrid Stepper Motor. The servo motors are powered by Arduino Uno board.

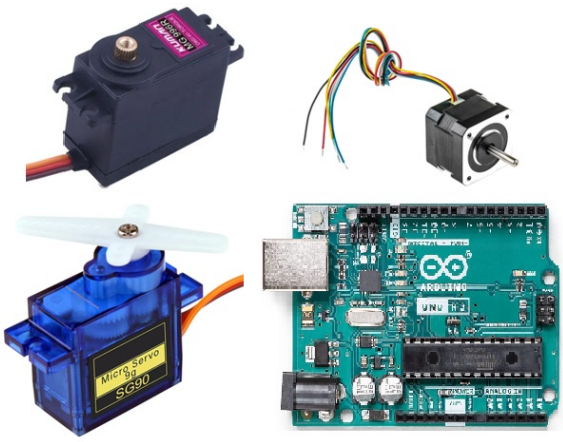

Figure 13: Arduino and Motors

6.3 Schematic Diagram

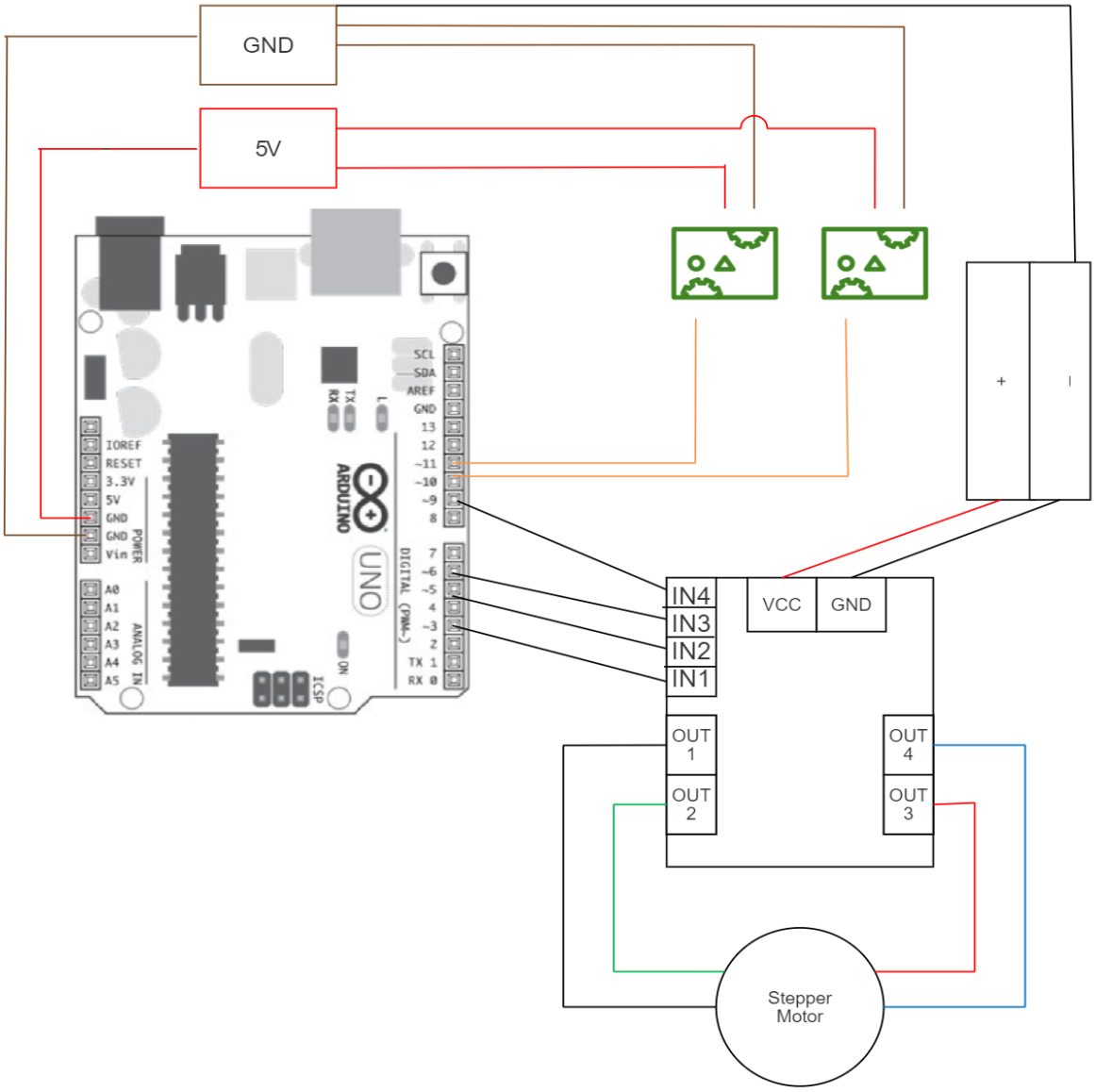

Figure 14: Schematics for Electrical

- The Arduino Uno R3 board is connected to a computer via USB Cable.
- The Micro Servo motor has 3 connections, Power Supply, Ground, and Input. Connect the Power supply and Ground to the Arduino Uno board. The input pin should be connected to one of the PWM pins in the Arduino.
- The RC Servo motor has similar connection as micro servo motor, it is also connected to the Arduino's power supply and ground. It uses a PWM pin as an input pin.
- The stepper motor is connected to the L298N motor driver. The motor driver has 4 OUT connections. The Phase A connections of stepper motor (Black and Green wires) are connected to OUT 1 and OUT2. The Phase B of stepper motor (Red and Blue wires) are connected to OUT3 and OUT4.

- The motor driver is powered by a 9V power supply that is directly connected to it. The motor driver has 4 IN pins that are connected to 4 PWM pins of the Arduino Uno board.

7 Image Processing

7.1 Treatment steps

The principle of this project is to draw a picture in one process, while using only one pen. To achieve that, it must be able to convert a shaded, coloured picture, into a binary, black-and-white picture. It also must be in a dot matrix format.

The best tool to execute such treatment is MATLAB. From an original image, there is an applied succession of filters that will realize the objective of the case. Figure 15 shows the MATLAB code for the image processing.

```
penGirth = 0.07; % Pen girth in cm
width = int16(7.85/penGirth); % number of colums drawn, 7.85 = pi*2.5
height = int16(10/penGirth); % number of lines

im_original = imread('astronaut.jpg'); % original image
figure, imshow(im_original);

im_grey = rgb2gray(im_original); % grey image
im_scaled = imresize(im_grey,[height, width]); % resized to fit the cylinder
im_sharped = imsharpen(im_scaled); % sharp the edges for better edge detection
edge_im = edge(im_sharped,'canny', 0.12);  % keep the edges, paramater to vary for each image for good results
edge_inv = ~edge_im;
figure, imshow(edge_inv);
```

Figure 15: Image Processing MATLAB code

The first filter applied is a greyscale filter, to convert the image to a greyscale one. Then, the image is resized. To define the dimensions of the resized image, the girth of the pen extremity was considered. The number of rows (height) and columns (width) to draw by dividing the dimensions of the cylinder plane (2.5π*10, in cm) by the pen girth was the obtained. Then, a sharpening filter is applied, which will facilitate the binary conversion in the next stage of the process.

To convert an image into binary black and white, the Canny edge filter was used. This filter takes a parameter which determines how much of the image will be converted to white relative to black. As the result of the filter is that the edges are in white and the rest is black, thus this result was inverted for practical reasons: the edges (being the most defining feature of the final image) will be what the robot will draw, therefore black. A diagram of the process is described in Figure 16.

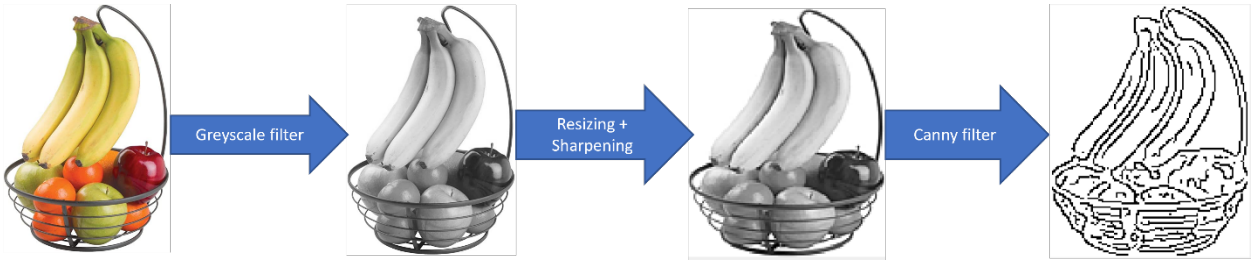

Figure 16: Diagram of the Image Treatment Steps

7.2 User Interface

To facilitate the usage of the program, a User Interface was designed in which it implemented the treatment described in the previous part. By launching the UI, this page opens:

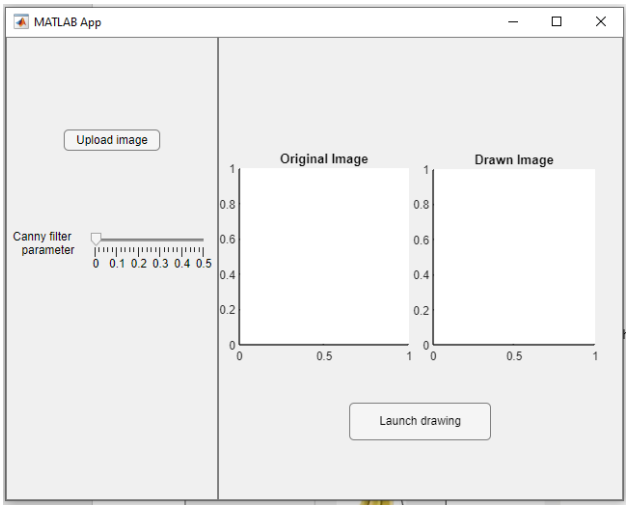

Figure 17: Landing Page of the Image Processing UI

The user can then choose the image they want to draw by clicking on the “Upload Image” button. By doing so, the image appears in the “Original Image” panel.

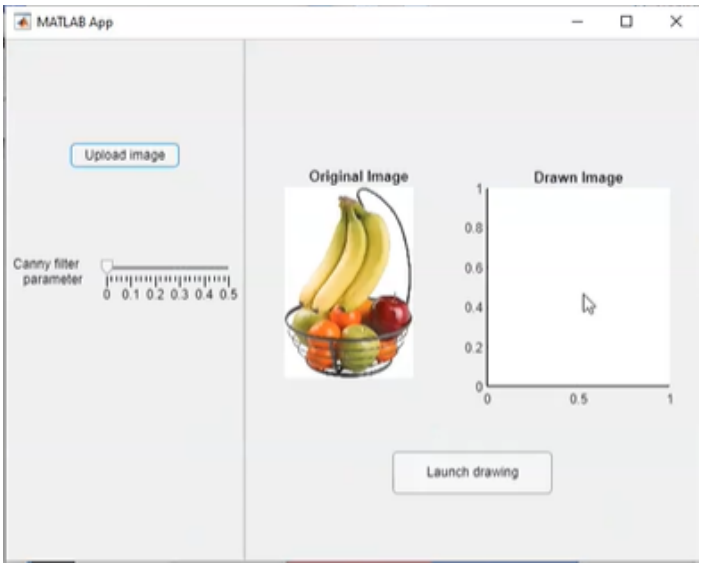

Figure 18: “Image Chosen” Page of the Image Processing UI

The User can then vary the value of the parameter of the edge canny filter, producing different results in the “Drawn Image” panel.

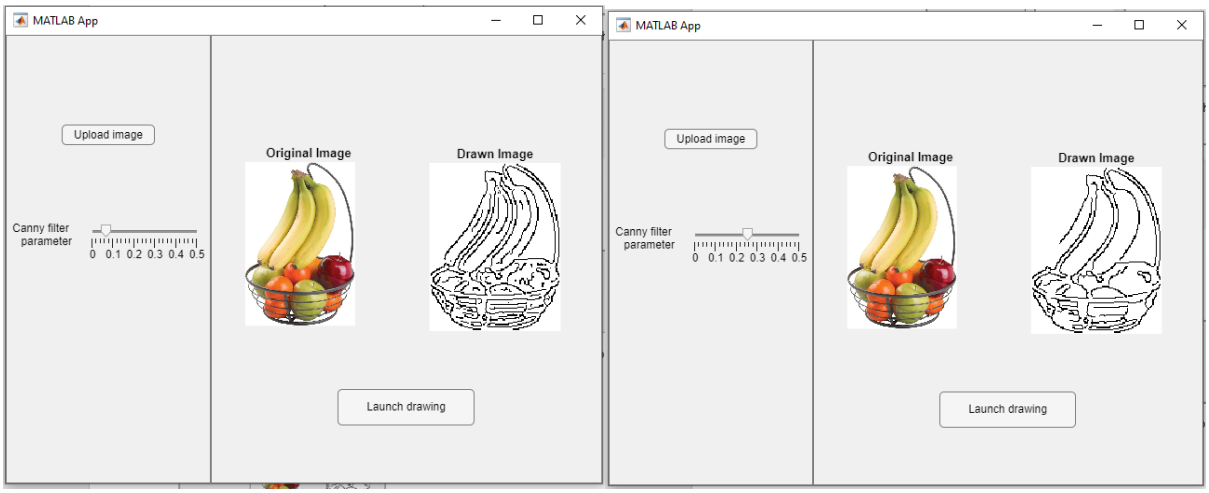

Figure 19: Two states of the Drawn Image Panel depending on the Canny Filter Parameter

When the binary image is satisfactory for the user, the user can then proceed to the drawing by clicking on the "Launch Drawing" button.

7.3 Potential for Improvement

There are a few ways to optimize this process of image treatment:

- The pen girth could be a parameter to vary with the UI, like the Canny filter parameter.
- The resizing of the image could consider the orientation of the original image: in our case, if our image is in landscape orientation (width > height), the resizing will compress the image to fit the cylinder, producing in result a heavily deformed image. The resizing could for example, if the image is more thick than high, rotate the image 90 degrees, so it would be painted sideways on the cylinder, but without applying such heavy deformation.

8 Transmission of the information

As described in the next part, the motor control will be handled with Arduino. For the Arduino program to obtain the information of the image to draw, it is needed to design a way of transmitting the image data from MATLAB to Arduino.

8.1 Inconclusive method: Serial Communication

The first method explored was to use the USB COM port by which the Arduino was connected to the computer to communicate the image (an array of 0 (black pixel) and 1 (white pixel)). This would ensure a direct communication without any user intervention.

After trailing, this method wasn't successfully implemented due to its complexity and the lack of time for studying the necessary methods for the approach. As the communication of data between the two programs wasn't the main objective of the project, it was chosen to manually communicate the data from MATLAB to Arduino.

8.2 Selected method: Manual copy/paste

A simpler way to transmit information is to copy the output of the MATLAB program directly into the Arduino program.

Nevertheless, when directly copying the final image raw data (the array of 0 and 1), the amount of data used by the Arduino program is often too large for it to handle. A more practical way is to transmit, rather than the image in a binary form, a list of the index of the black pixels to draw.

This is achieved by the piece of code placed after the image processing part in MATLAB:

```
% Formating for Arduino syntax
n_pixels = sum(edge_inv(:) == 0);
k = 1;
id_pixels = zeros([n_pixels, 2]);
for i = 1:n_cols
    for j = 1:n_rows
        if (edge_inv(j, i) == 0)
            id_pixels(k, 1) = i;
            id_pixels(k, 2) = j;
            k = k+1;
        end
    end
end
writematrix(id_pixels, "id_pixels.txt");
```

Figure 20: MATLAB code for Data Formatting

This code goes through the final image binary array and every time it encounters a black pixel (the array's element equals 0), it adds the index in the form (column, row) to another 1D array, for which a .txt file is created. An example of such a file is shown in Figure 21.

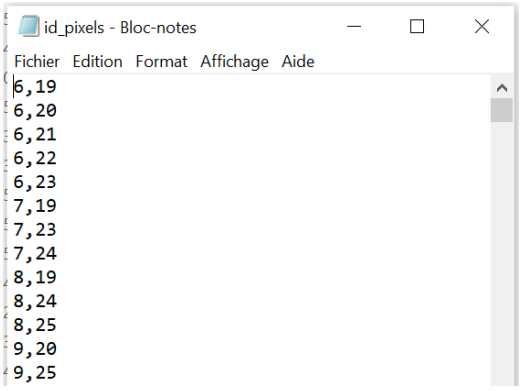

Figure 21: Example of list of pixels index.

The content of this file is then copied, reshaped to fit the Arduino syntax, and used as an input for the motor control program.

9 Motor Control

The control of the two servomotors and the stepper motor is handled through an Arduino program. Each of the parts are detailed below:

```
#include <Servo.h>
#include <Stepper.h>
#include <stdbool.h>

Servo servo_cyl;
Servo servo_pen;

const int stepsPerRevolution = 200;
Stepper stepper_belt(stepsPerRevolution, 3, 5, 6, 9);;


int pos_pen_on = 90;
int pos_pen_off = 150;
int pos_cyl_ini = 0;
int pos_cyl_actual = pos_cyl_ini;
```

Figure 22: Initialization Part of the Motor Control Arduino Code

In this part, the libraries used were imported. These libraries include ones related to the servomotors, the stepper motor, and the use of Booleans. The motors objects were then declared. stepsPerRevolution is the variable defining how many steps in one complete revolution (360 degrees) the stepper motor will execute. Then, the two positions for the servomotor handling the pen application, the off position (off the cylinder) and on position (on the cylinder) are defined.

```
void setup() {
  // put your setup code here, to run once:
  short int id_pix[391][2] =
{{2,4},  {2,5},  {2,6},  {2,8},  {2,9},  {2,16}, {2,17}, {2,20},  {2,21}, {2,26}, {2,30},  {2,41}, {2,43}, {2,44}, {2,45}, {3,3},  {3,
{6,49}, {7,2},  {7,3},  {7,7},  {7,8},  {7,34}, {7,35}, {7,39}, {7,40},  {7,44}, {7,45}, {7,46}, {7,47}, {8,13}, {8,14}, {8,32}, {8,33
{11,26},  {11,35},  {11,47},  {11,48},  {11,49},  {12,9}, {12,27},  {12,44},  {12,46},  {12,47},  {13,8}, {13,9}, {13,27},  {13,28},
{17,12},  {17,15},  {17,20}, {17,25},  {17,28},  {17,36},  {17,37},  {18,8}, {18,12},  {18,15},  {18,20}, {18,25},  {18,26},  {18,27},
{21,38},  {21,39},  {21,44},  {22,13},  {22,14},  {22,15},  {22,19},  {22,20}, {22,25},  {22,31},  {22,39},  {22,40}, {22,41},  {22,42
{26,31},  {26,32},  {26,33},  {26,43},  {27,16},  {27,19},  {27,20}, {27,33},  {27,34},  {27,35},  {27,36},  {27,37},  {27,38},  {27,4
{34,9}, {34,13},  {34,24},  {34,25},  {34,29},  {34,35},  {34,40}, {34,41},  {34,42},  {34,43},  {35,7}, {35,12},  {35,13},  {35,20},

  servo_cyl.attach(11);
  servo_pen.attach(10);

  stepper_belt.setSpeed(40);

  servo_pen.write(pos_pen_off);
  servo_cyl.write(pos_cyl_ini);
```

Figure 23: Set-up Part of the Motor Control Arduino Code

In the set-up part of Arduino (executed only once), the list of indexes of pixels to draw was copied from the MATLAB output file. The pin on which the servos are attached are declared, thus the speed of the stepper motor and the initial positions for the cylinder and the pen are set.

```
int prev_pos_x = 1;
int prev_pos_y = 1;
for (int i = 0; i<391; i++)
{
    if(id_pix[i][0] != prev_pos_x) // if we change column
    {
        pos_cyl_actual = pos_cyl_actual + (id_pix[i][0]-prev_pos_x)*(180/39); // rotating cylinder accordingly
        servo_cyl.write(pos_cyl_actual);
        prev_pos_x = id_pix[i][0];
        delay(1000);
    }
    int diff_y = id_pix[i][1]- prev_pos_y;
    // Moving along the Y axis
    stepper_belt.step(int(stepsPerRevolution*diff_y*0.016)); // value 5 to change
    prev_pos_y = id_pix[i][1];
    delay(500);
    servo_pen.write(pos_pen_on); // Applying pen
    delay(500);
    if (i < 390 && (id_pix[i][1]+1) != id_pix[i+1][1]) // If next pixel to draw not directly after
    {
        servo_pen.write(pos_pen_off); // removing pen
        delay(500);
    }
    delay(500);
}
servo_pen.write(pos_pen_off);
```

Figure 24: Main Part of the Motor Control Arduino Code

Figure 24 details the main section of coding needed for the robot to work. Two variables, prev_pos_x and prev_pos_y, will be used to store the X and Y coordinates of the previous drawn pixel. These are initialized at 1.

A loop then goes through every index. For each index, there are four main parts of treatment:

1. First, it is checked if the column of the pixel to draw has changed since the last drawn pixel, using the prev_pos_x variable. If true, the cylinder is rotated to go through the current column and update the prev_pos_x variable.
2. Then, the difference in height (Y axis) of the current pixel in terms the last drawn pixel is calculated. The stepper motor moves the pen along the Y-axis accordingly.
3. Then, the pen is applied on the cylinder, drawing the pixel.
4. Finally, it is checked if the next pixel to draw is located directly below the current pixel. If true, there is no need to remove the pen but if this returns as false, the pen is then removed.

When every pixel has been drawn, the pen is removed from the cylinder and the program is completed.

10 Accessory program: Trajectory planning

To include in this case study one topic seen in the course we chose to implement an additional MATLAB program realizing trajectory planning using our robot's configuration. The goal is to, from the original coordinates of a simple shape (here a lightning bolt), realize trajectory planning calculus and plot the result in 3D. Figures 25 to 31 explain the program's structure.

```
D_p = 4; % diameter of pulleys
d_intp = 20; % distance between pulleys center
D_cyl = 5;

X_points=[7 7 6 6.2 4 4.2 3 2 3.1 2.8 5.5 5.2 7 7]; % x coordinates of lightning points
Y_points=[4 4 5 5.1 6.6 6.7 8 7.2 6.5 5.7 4.5 4.2 4 4]; % y coordinates of lightning points
Z_points=[1 0 0 0 0 0 0 0 0 0 0 0 0 1];
%X_points = [1 3 7];
%Y_points = [4 2 2];

plot3(X_points, Y_points, Z_points, 'r','LineWidth',6);
hold on;
```

Figure 25: Initialization Part of the Trajectory Planning MATLAB code

First, the robot's mechanical parameters are defined, and then, the points of the lightning bolt in 3D, relative to the cylinder surface space. By including the initial and final position where the pen is not on the cylinder, indicating the beginning and ending points of the trajectory.

```
tf = 1;
t = 0:0.1:tf;

theta_c = X_points./(D_cyl/2);
theta_l = Y_points./(D_p/2);
theta_p = asin((0.5+Z_points)./1.6);
```

Figure 26: Temporal and Angle Definition Part of the Trajectory Planning MATLAB code

Then, temporal parameters are defined, as well as the values of the angles for each point.

The cylinder angle is directly a transform from a circle's circumference increment to an angle, using an inverse proportionally relation with the circle's radius, here being the cylinder radius.

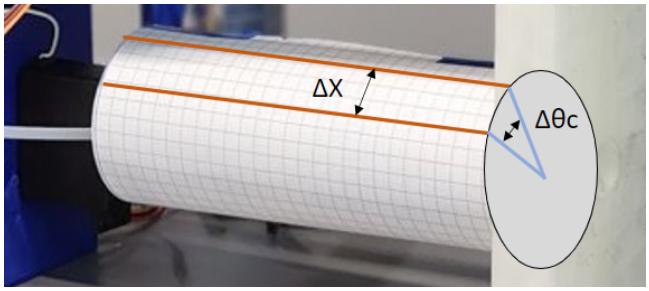

Figure 27: Cylinder angle to translation transformation in a scheme

Same principle applies for the angle of the stepper motor for the belt:

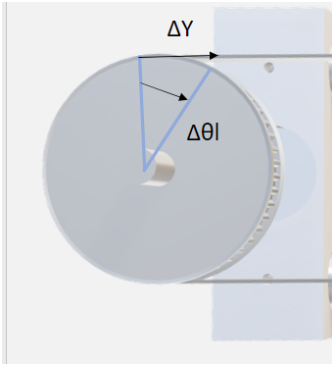

Figure 28: Pulley angle to translation transformation in a scheme

The system regarding the pen can be represented with the following scheme:

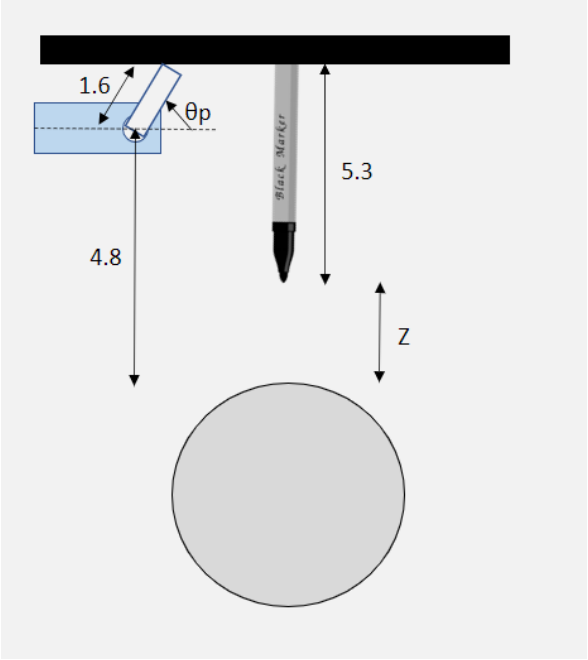

Figure 29: Pen distance to the cylinder depending on the servo angle in a scheme.

The expression was derived as the following formula:

$$\sin(\theta_p) = \frac{0.5 + Z}{1.6}$$

```
traj_theta_c = zeros(size(theta_c, 2)-1, size(t, 2));
traj_theta_l = zeros(size(theta_l, 2)-1, size(t, 2));
traj_theta_p = zeros(size(theta_p, 2)-1, size(t, 2));

Px = [];
Py = [];
Pz = [];

for i= 1:size(traj_theta_c, 1)
    traj_theta_c(i,:)= theta_c(i)+3/(tf^2).*(theta_c(i+1)-theta_c(i)).*(t.^2)+(-2/(tf^3)).*(theta_c(i+1)-theta_c(i)).*(t.^3);
    traj_theta_l(i,:)= theta_l(i)+3/(tf^2).*(theta_l(i+1)-theta_l(i)).*(t.^2)+(-2/(tf^3)).*(theta_l(i+1)-theta_l(i)).*(t.^3);
    traj_theta_p(i,:)= theta_p(i)+3/(tf^2).*(theta_p(i+1)-theta_p(i)).*(t.^2)+(-2/(tf^3)).*(theta_p(i+1)-theta_p(i)).*(t.^3);
    Px = [Px traj_theta_c(i,:).*(D_cyl/2)];
    Py = [Py traj_theta_l(i,:).*(D_p/2)];
    Pz = [Pz (sin(traj_theta_p(i,:)).*1.6)-0.5];
end

plot3(Px, Py, Pz, '-o', 'Color', 'g', 'MarkerSize', 5, 'LineWidth',2);
```

Figure 30: Main Part of the Trajectory Planning MATLAB code

Finally, the trajectories for each angle using the single cubic polynomial formula:

$$\theta(t) = \theta_i + (3/t_f^2)(\theta_f - \theta_i)t^2 + (-2/t_f^3)(\theta_f - \theta_i)t^3$$

Trajectory points were calculated, based on the equations derived beforehand. The final plot is the following:

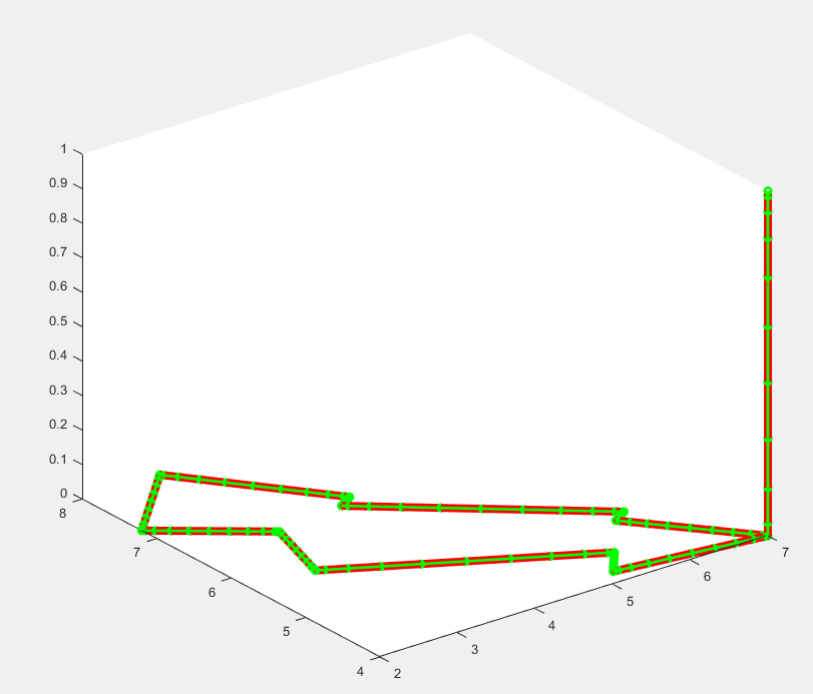

Figure 31: Comparison Figure of the Trajectory Planning MATLAB code

In this case, all motors handle one precise coordinate (X for the cylinder servo, Y for the stepper, Z for the pen servo), and are all independent between the three (the Z-axis of the pen servo is not affected by the Y-axis stepper). Hence, contrary to a multi-joint robotic arm, for example, there are no errors between the objective and the drawn trajectory.

11 Testing and Results

11.1 Results

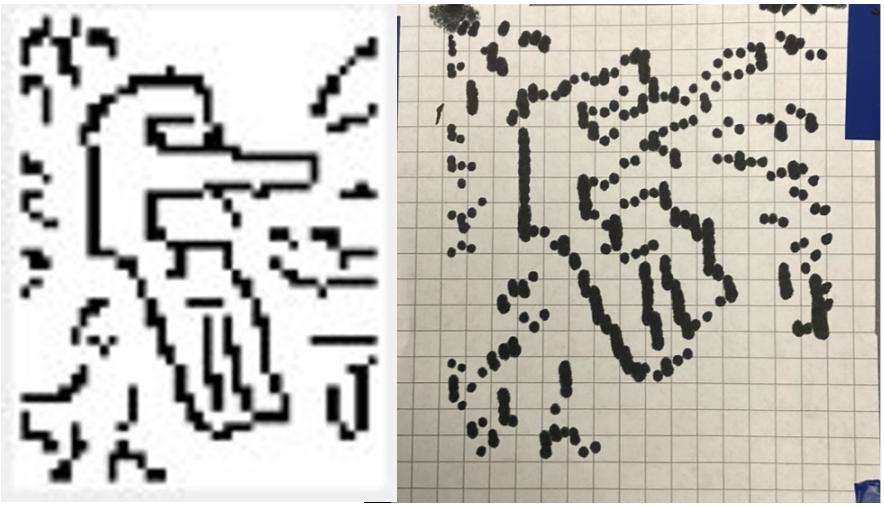

Figure 32: Comparison of the output pixelated picture (left) and the output drawing (right)

In Figure 32 above, the results can be seen for the Case Study. The robot was able to produce the drawing but didn't have the accuracy or precision which was needed. With some tweaking to the code and to the design, this can be optimized for the best possible image to be printed. Another reason for the inaccuracy was mechanical. Future work would be to re-design the robot to make sure all the 3D printed parts were up to a higher standard. On top of this, the robot could be streamlined and use better materials to ensure nothing goes wrong. For example, the pen needed a plate so the motor could lift it and lower it. For that plate, cardboard was used. This could be upgraded to a 3D printed plate as Cardboard is flimsy and doesn't necessarily keep its shape.

11.2 Challenges or Limitations

As mentioned previously in Final Design Description some parts were manufactured using 3D printing due to customization requirements. Using 3D printers, the goal of obtaining customized components was met. However, the resolution from the printing had a +0.3mm offset. This then resulted in several parts not fitting perfectly. For example, the component that required the highest resolution was the synchronous pulley. The printed teeth were not defined enough to hold the synchronous belt in place, which caused the belt to slip whilst the stepper motor tried to rotate. Hence, for some parts, manual finishing or reprinting was required to compose the best fit for the parts. Printing time totalled up to be 12 hours, including parts that required reprinting due to failures and dimension errors.

The materials used were wood and PLA filaments. The wooden rod was used as guiding rods to support the pen holder as well as lead the pen holder to only move on laterally. The wooden rods were not completely straight, and surface was rough, causing the system to sometimes stop due to lack of torque from the slipping belt.

Cable management was a limitation to the robot. Because the cables were not kept neat and orderly, they became hard to work with and did not provide the freedom that was required to the robot and the pen holder. In the future, proper cable management will be implemented to ensure maximum optimization.

The drawing quality of the robot was inaccurate. This was due to slipping in the gears as well as the incorrect movement of the motor connected to the cylinder. To fix this the gears must either be manufactured using metal, or 3D printed using a lower error. The movement of the motor can be updated and changed so the cylinder is moved precisely. Another way to ensure higher drawing quality would be to use a different pen that has a finer point. This could be combined with using a higher resolution of points to ensure better quality.

12 Conclusion

In conclusion, the case study details the design, fabrication, assembly, and testing of a drawing robot designed based on the theory of a dot matrix printer. The robot incorporated many of the main elements taught in this course including kinematic modelling, MATLAB, and CAD. This ensured that the robot was well-rounded, delivering upon the objective to create a drawing robot that could draw on a cylinder.

13 Reference

1.) M. J. Allen, “Dot-matrix printers,” *Forensic Science International*, vol. 35, no. 4, pp. 283–295, 1987.

2.) C. W. White, E. M. Brussell, T. T. Williams, and S. J. Eog, “Gray-scale graphics using dot matrix printers,” *Behavior Research Methods, Instruments, & Computers*, vol. 16, no. 3, pp. 273–276, 1984.

3.) B. R. Sundheim, “Graphics with a dot-matrix printer,” *Journal of Chemical Education*, vol. 61, no. 6, p. 531, 1984.

14 Appendix

CAD drawing of the parts:

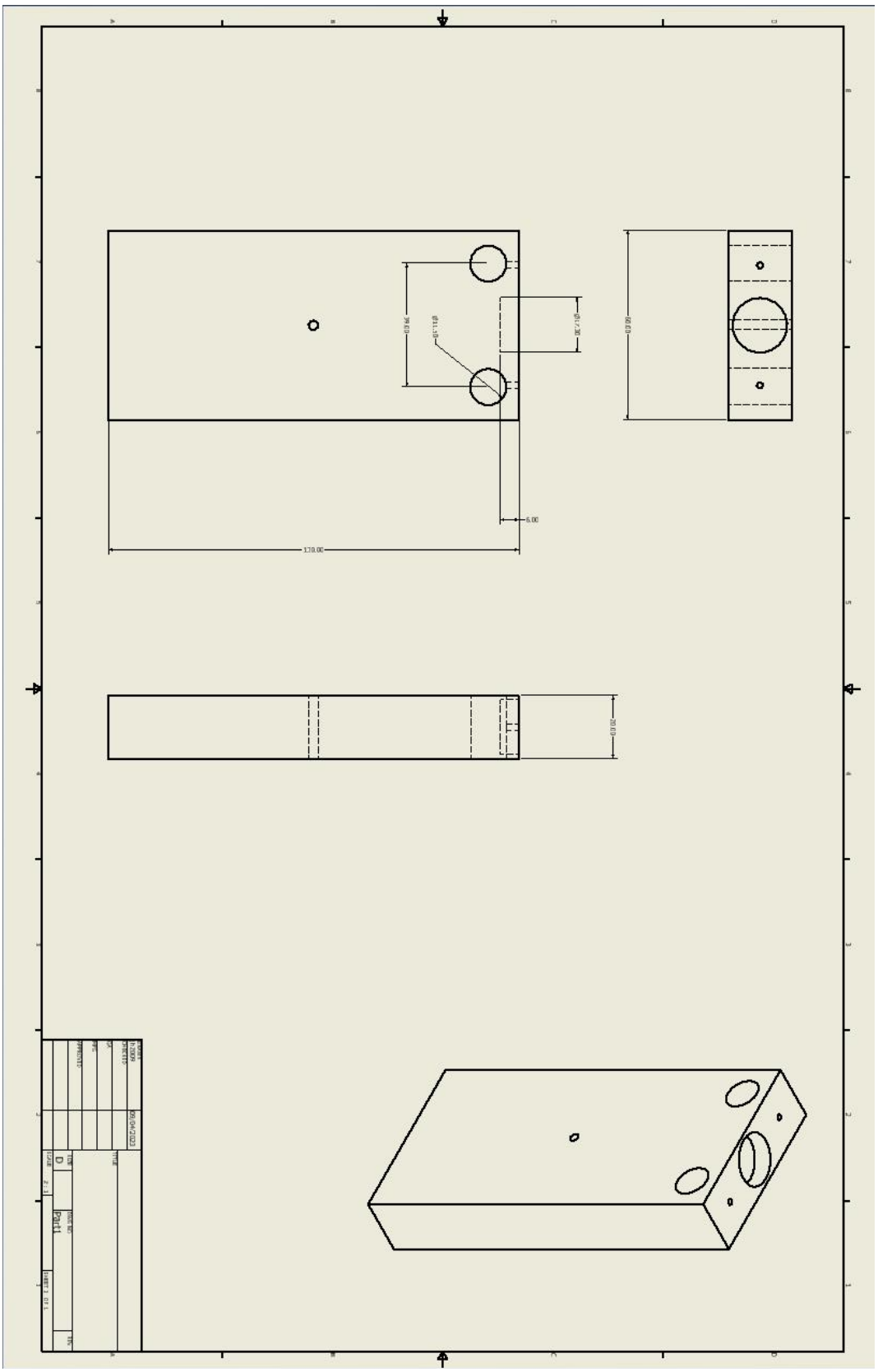

Figure 33: DWG for side stands

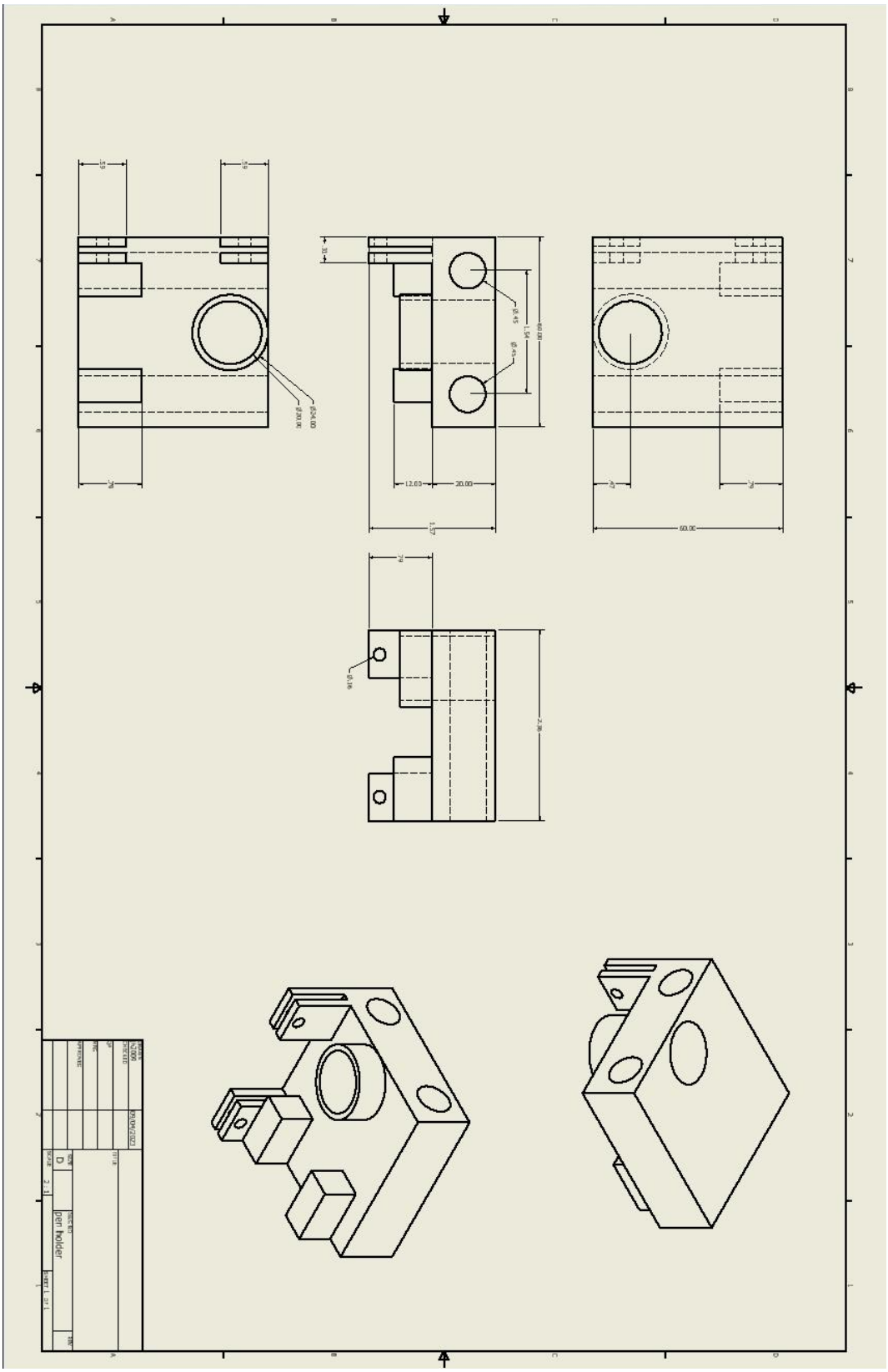

Figure 34: DWG of pen holder

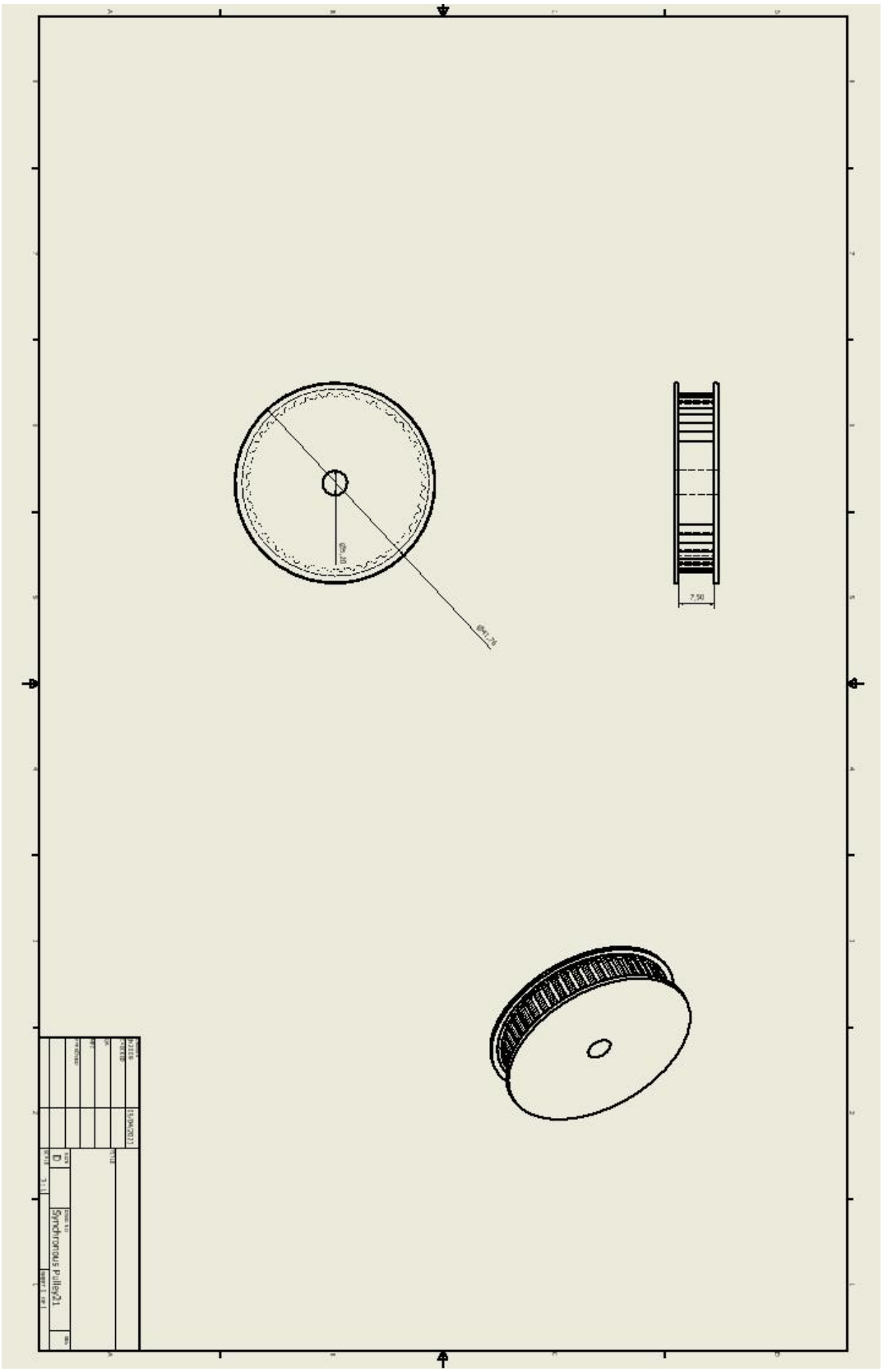

Figure 35: DWG of Synchronous Pulley

15 Annex

Course code and name:	B50RO – Robotic Mechanical Systems 2022-2023
Type of assessment:	Group
Coursework Title:	Case Study Report
Student Name:	Jijing Hon
Student ID Number:	H00358337

Declaration of authorship. By signing this form:

- **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section.

- I confirm that I have read, understood and followed the University's Regulations on plagiarism as published on the University's website, and that I am aware of the penalties that I will face should I not adhere to the University Regulations.

- I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on Academic Integrity and Plagiarism

Student Signature (type your name): Jijing Hon

Date: 24/04/2023

Course code and name:	B50RO – Robotic Mechanical Systems 2022-2023
Type of assessment:	Group
Coursework Title:	Case Study Report
Student Name:	James Whitehouse
Student ID Number:	H00336363

Declaration of authorship. By signing this form:

- **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section.

- I confirm that I have read, understood and followed the University's Regulations on plagiarism as published on the University's website, and that I am aware of the penalties that I will face should I not adhere to the University Regulations.

- I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on Academic Integrity and Plagiarism

Student Signature (type your name): James Whitehouse

Date: 24/04/2023

Course code and name:	B50RO – Robotic Mechanical Systems 2022-2023
Type of assessment:	Group
Coursework Title:	Case Study Report
Student Name:	Pranjal Samant
Student ID Number:	H00425492

Declaration of authorship. By signing this form:

- **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section.

- I confirm that I have read, understood and followed the University's Regulations on plagiarism as published on the University's website, and that I am aware of the penalties that I will face should I not adhere to the University Regulations.

- I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on Academic Integrity and Plagiarism

Student Signature *(type your name):* *Pranjal Samant*

Date: 24/04/2023

Course code and name:	B50RO – Robotic Mechanical Systems 2022-2023
Type of assessment:	Group
Coursework Title:	Case Study Report
Student Name:	Clément Noblet
Student ID Number:	**H00428424**

Declaration of authorship. By signing this form:

- **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section.

- I confirm that I have read, understood and followed the University's Regulations on plagiarism as published on the University's website, and that I am aware of the penalties that I will face should I not adhere to the University Regulations.

- I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on Academic Integrity and Plagiarism

Student Signature *(type your name):* *Clément Noblet*

Date: 24/04/2023

Course code and name:	B50RO – Robotic Mechanical Systems 2022-2023
Type of assessment:	Group
Coursework Title:	Case Study Report
Student Name:	Hugo Millet
Student ID Number:	H00419579

Declaration of authorship. By signing this form:

- **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section.

- I confirm that I have read, understood and followed the University's Regulations on plagiarism as published on the University's website, and that I am aware of the penalties that I will face should I not adhere to the University Regulations.

- I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on Academic Integrity and Plagiarism

Student Signature *(type your name):* *Hugo Millet*

Date: 24/04/2023