ELEC60030
Robotic Manipulation
Coursework Report 2022

Jayjun Lee

Contents

1 Task 1. Modelling and Simulation

1.1 Introduction

The objective of this coursework is to model the Open Manipulator X robotic arm, to simulate the model's forward kinematics and analytically solve the inverse kinematics to use it for further control of the robot arm in a desired sequential trajectory of motion. This section deals with the modelling and simulating the robot.

1.2 Open Manipulator X Robot

The robotic arm's dimensions are given as Figure 1.1, which are used to model the robotic arm on `MATLAB`. When modelling the robot, the links that correspond to the lengths of 0.077, 0.130, 0.124 and 0.126 m are used. The diagonal link of length 0.130 m is used to simplify the model and its inverse kinematics, although a post-processing step to remove the extra angle introduced by such simplification is added and explained later.

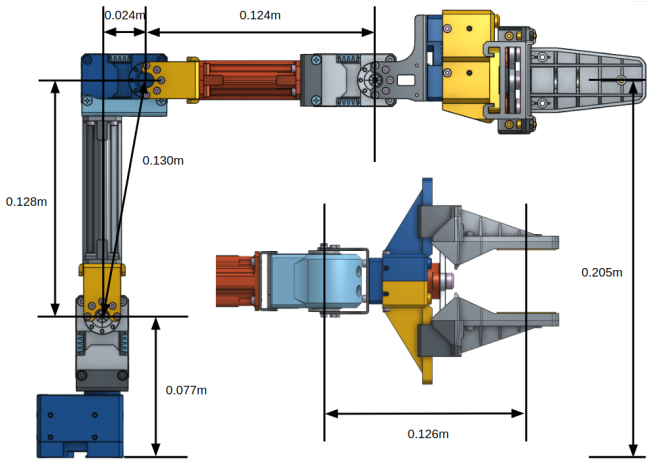

Figure 1.1: Schematics Diagram of the Open Manipulator X Robot and its Dimensions. [1]

1.3 Coordinate Frame Assignment

A world frame at the base of the robot and 4 additional coordinate frames are assignment at each joint of the robot arm for modelling and analysis as in Figure 1.2. The world frame defines the relative positions of other coordinate frames by transformation matrices. When modelling the robot, it is essential to include the information of each joint's pose and orientation that a coordinate frame is assigned such that the x-axis is along the link length, z-axis is along the joint axis and y-axis follows the right hand rule. This format is used to apply Denavit-Hartenberg (DH) convention and notation.

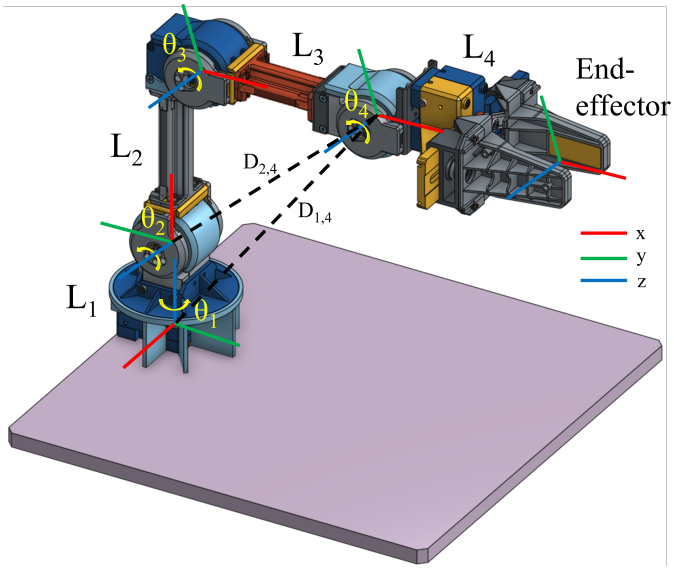

Figure 1.2: World base frame and joint coordinate assignment diagram on the Open Manipulator X robot with other parameters used for deriving the inverse kinematics such as $D_{1,4}$ and $D_{2,4}$. (only moving parts are drawn, not base parts)

1.4 DH Notation Table and Transformation Matrices

Following the world frame and coordinate frames assignments, the following DH notation table is deduced as shown in Table 1.1.

Table 1.1: DH Parameters for the Open Manipulator X Robot Links.

Link i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	L_1	θ_1
2	0	$\frac{\pi}{2}$	0	θ_2
3	L_2	0	0	θ_3
4	L_3	0	0	θ_4
5	L_4	0	0	0

From Table 1.1, the values for each parameter are used to generate the transformation matrices from one link to the next link by following Equation 1.1. The entire transformation matrix from the base of the robot to the end-effector point can be represented as in Equation 1.2.

$$T_{\text{DH}} = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & a_{i-1} \\ \sin(\theta_i)\cdot\cos(\alpha_{i-1}) & \cos(\theta_i)\cdot\cos(\alpha_{i-1}) & -\sin(\alpha_{i-1}) & -\sin(\alpha_{i-1})\cdot d_i \\ \sin(\theta_i)\cdot\sin(\alpha_{i-1}) & \cos(\theta_i)\cdot\sin(\alpha_{i-1}) & \cos(\alpha_{i-1}) & \cos(\alpha_{i-1})\cdot d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.1}$$

$$^{0}_{5}T = {}^{0}_{1}T\ {}^{1}_{2}T\ {}^{2}_{3}T\ {}^{3}_{4}T\ {}^{4}_{5}T \tag{1.2}$$

1.5 Forward Kinematics and Verification of DH Parameters

Equation 1.2 defines the forward kinematics of the robotic arm model where if the angles are given, the robotic arm will be in that orientation. The DH parameters are verified by incrementing each angle slightly by comparing it with the expected change and the orientation of the robotic arm in simulation. Our code also ensures the matrices are correct by asserting the constant link lengths. The verification of the working solution via forward kinematics is included in the demonstration video that displays the movement for initialising to the start position as in Figure 1.3.

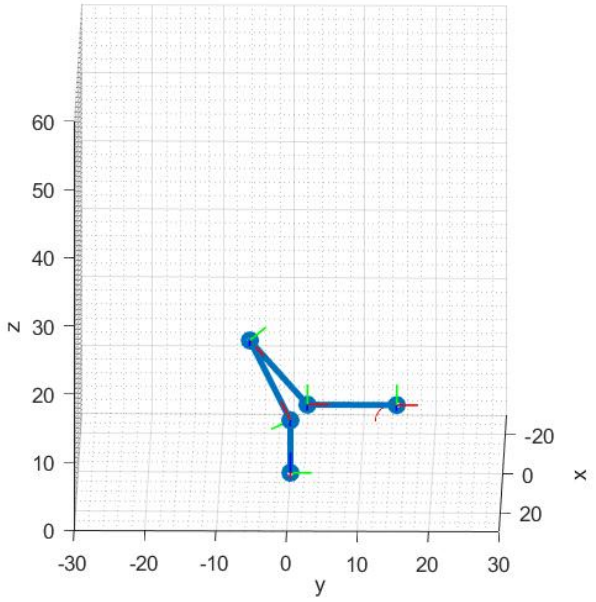

Figure 1.3: Simulation of Task 1 forward kinematics to initialise the robot to its starting position.

1.6 Analytical Solution of Inverse Kinematics

The derivation of the inverse kinematics of the robotic arm are shown below. The system is an under-determined system and can yield multiple solutions for each desired endpoint. Multiple solutions are yielded due to the Pythagoras Theorem being used where the displacements may be positive or negative as in Equations A.6 and A.7 in Appendix . To add on, the use of cosine rule, which always finds the acute angle when given the sides, also leads to multiple solutions in A.8 and A.9. Therefore, the end-effector angle has to be specified depending on the desired orientation of the gripper. It is important to split such cases depending on the desired end-effector angles. Most of the time, the parallel grip to the ground and downward grip facing the ground are used from Equation A.1 for various purposes such as rotating and transferring objects into different orientations. The displacements $D_{1,4}$ and $D_{2,4}$ in Equation A.6 and A.7 indicate the displacements between the first and the fourth joints and the second and the fourth, respectively, which are shown in Figure 1.2.

1.7 Simulation Results

Figure 1.4, shows the end result of the simulation that draws a 10 by 10 cm squares in each Cartesian plane. The actual simulation is included in the video.

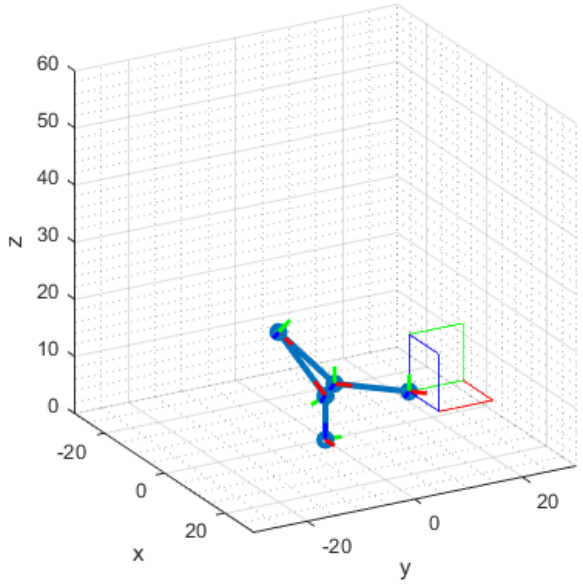

Figure 1.4: Simulation of Task 1 of drawing 10 cm by 10 cm squares in each Cartesian plane.

2 Task 2. Pick and Place

2.1 Mapping Angles from Inverse Kinematics to Encoder Values

2.1.1 Calculation of Neglected Angle Offset on the Simplified Link 2

Between joints 2 and 3, there exists an offset angle due to their joint alignments that has to be taken into account when performing forward kinematics on the robot as this was not taken into account in our simulation model for simplifying purposes. The angle can be calculated as such.

$$\theta_{\text{offset}} = \tan^{-1}\left(\frac{0.024}{0.128}\right) = 0.1853 \text{ [rad]} = 10.62 \text{ [deg]} \tag{2.1}$$

2.1.2 Conversion from Inverse Kinematics Angles to Encoder Values

Following the frame assignment in Section 1, the angles defined for each joint are different. This is because for every joint, there are two frames: the base part and the moving part of the joint. In Figure 1.2, the base parts are not indicated for clearer image. However, due to these different definitions of each joint, different offsets need to be applied for each before converting to the motor encoder values to control the smart servo motors.

$$\theta_1' = [\theta_1 + \frac{\pi}{2}] \cdot \frac{4096}{2\pi} \tag{2.2}$$

$$\theta_2' = [\frac{3\pi}{2} - (\theta_2 + \theta_{\text{offset}})] \cdot \frac{4096}{2\pi} \tag{2.3}$$

$$\theta_3' = [2\pi - \theta_3 + \frac{\pi}{2}] \cdot \frac{4096}{2\pi} \tag{2.4}$$

$$\theta_4' = [\pi - \theta_4] \cdot \frac{4096}{2\pi} \tag{2.5}$$

2.2 Waypoint Trajectory Generator given Start and End Points

The waypoint generator generates the trajectory from the start to the end cube positions. Two cube positions – one for the start and the other for the end – are inputted to the generator where the cube position consists of the 3D coordinate of (x,y,z) and the desired end-effector angle, ϕ.

In order to generate a smooth trajectory, the following sequences and motions have to be taken into account: the gripping and ungripping, picking up and putting down, transferring from one position to the other position all with different end-effector angles.

The generator works in the following order. It generates coordinates that start from above the start cube position with the grip open. Then, the gripper moves down to the cube's height and grips. The gripper returns to the position above the cube with the grip on and then transfers to the position that is above the end cube position's height. With the grip still on, it moves down to the height of the end cube and then ungrips the gripper and returns to the position above the cube to end its sequence.

At each waypoint, the robot will follow the desired end-effector angle to complete each subtrajectory.

2.3 Task A: Transferring Cubes from Starting to Finishing Positions

The layout and positions of the cube holders are shown in Figure 2.1. Task A was solved by transferring cube 2 to cube 5 position, cube 1 to cube 6 position and then cube 3 to cube 4 to finish. Therefore, 3 trajectories were generated from cube 2 to 5, cube 1 to 6 and cube 3 to 4 using the waypoint trajectory generator. The video of the working solution are included in the demonstration video.

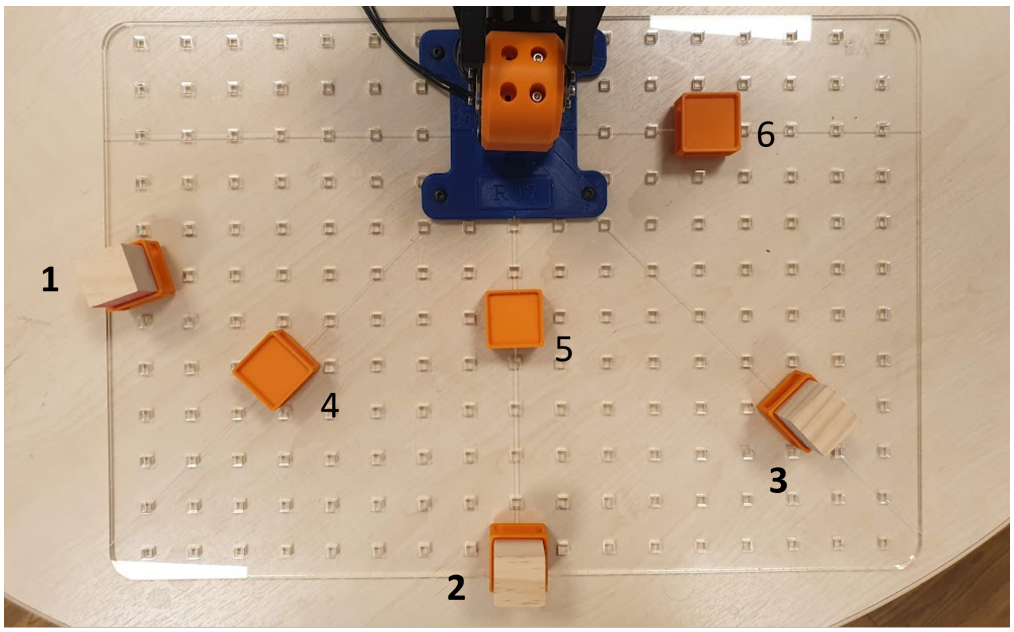

Figure 2.1: The robot's workspace and the cube holder layout indicating start (1, 2 and 3) and end (4, 5 and 6) positions.

2.4 Task B: Picking and Rotating Cubes to have Red Side at the Top

In Task B, the cubes were set out by putting the cube 1 upside down such that the red side faces the ground, putting the red side of the cube 2 outward from the robot and putting the red side of the cube 3 inward to (facing) the robot.

For the cubes to have their red sides at the top, they have to be rotated and this is achieved by approaching the same cube position but with different end-effector angles. For cube 2, the robot picks up the cube using the downward gripper angle and returns the cube using the parallel gripper orientation. For the cube 1, the same sequence for cube 2 are down twice to achieve the 180 degrees rotation. For cube 3, the cube is picked up in parallel orientation of the gripper first, then put down in the downward gripper orientation. The video of the working solution are included in the demonstration video.

2.5 Task C: Stacking all Cubes in a Finishing Position with Red Sides Facing Away

Task C follows the same initial settings as in Task B. For cube 2, the red side is already facing outwards that the trajectory is generated such that the robot picks it up and puts it down both in the downward orientations. For cube 3, it is first rotated by picking it up using the parallel gripper then put down in the same position in the downward orientation. Then the cube is picked up in parallel orientation again and then transferred to cube 5 position above cube 2, which is already there, by picking it up in parallel orientation and putting it in downward orientation, At last, the cube 1 is rotated first by picking it up in downward orientation then put back in parallel orientation to make its red side face outwards. Then the robot transfers the cube by picking it up and putting it down above cube 3 in cube 5 position using the downward gripper orientation. The video of the working solution are included in the demonstration video.

3 Task 3. Trajectory Following and Drawing

3.1 Open-Manipulator Robot Gripper 3D Design using CAD

The pen gripper for the trajectory following task is a printed 3D design from CAD using SolidWorks, shown in Figure 3.1. The process to design the gripper involves a left and a right part as shown below in two different orientations each. We designed the gripper to resemble a claw that fits the diameter of then pen of about 16mm. This prevents the pen from slipping in the radial direction. We also implemented fillet in the inside of the claw, as this reduces stress on the connection between the features and rounds off the claw edge to enable the pen to lifted smoothly as shown in Appendix B.1. The reason for having 2 fingers on the left claw and and 1 finger on the right claw is to prevent any rotational motion when grabbing the pen that may be incurred when using just a single finger for each claw.

Figure 3.1: Actual photo of the pen gripper holding a pen.

3.2 Waypoint Trajectory Generator

The shapes to follow and draw in Task 3 consists of picking up the pen and drawing out three lines and an arc as shown in Figure 3.2. In order to carry out these drawing motions, the following generators are designed. Then the inverse kinematics is used to follow the trajectory from the generators.

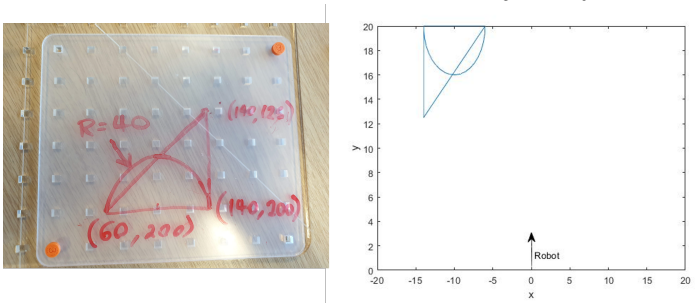

Figure 3.2: Three lines and an arc drawing for the robot to follow.

3.2.1 A Line Segment

The waypoints that form a line segment set out by two given end-points are generated by finding the linear space in the x-coordinates between the two points and also in the y-coordinates. The number of waypoints used to represent that line segment is also given and the waypoints of five are used as the default value, which yielded a well-drawn straight line.

3.2.2 An Arc Segment

The waypoints that form an arc segment set out by the two given points, the centre of circle, the radius and the start and end angles in radians. These are generated by using the polar coordinates using the centre of radius to translate the arc into the right position where the arc shape is drawn by the radius and the range of angles that it covers from start to end.

3.2.3 Pen Pick-up Sequence

For the pen pick-up sequence, similarly to the cube pick-up sequence in Task 2, the gripper hovers to the position above the pen with the grip open in downward orientation. Then it approaches the pen to grab it and returns to the position above the pen but in the parallel orientation of the gripper to the ground, indicating the completion of the pen pick-up sequence.

4 Task 4. Own Task

4.1 Task Description

For Task 4, we decided to perform a continuation of the previous task, which is to use the robot to do a repetitive task of cleaning and wiping off the drawing from Task 3. The task that we have decided to perform can be summarised in a series of steps below.

1. Pick up an alcohol sanitiser.
2. Squeeze a drop of the sanitiser on the drawing board.
3. Place back the sanitiser.
4. Pick up an eraser.
5. Erase the drawing with the eraser.
6. Place back the eraser.

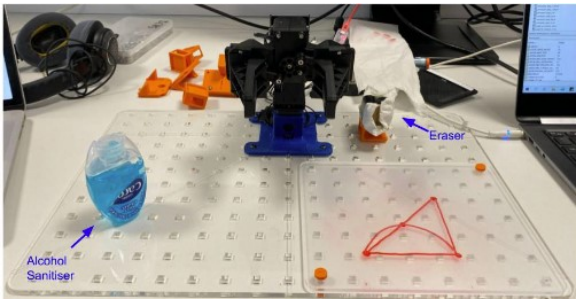

Figure 4.1: Setup for Task 4.

4.2 Motivation

The motivation between this task is that we as a team was tired of erasing the drawings done by the robot in Task 3 and figured that if the robot could clean the board itself, we would save a lot of effort and inky hands after every test run for Task 3. We also chose this task because it involves new movements by the robot that we have not implemented yet, such as variably squeezing a bottle such that a certain amount of 'liquid' comes out of the bottle, and pressing on the board with the eraser and conducting a sweeping motion. This could also be further used for other industrial applications or robot learning systems where the robot could itself learn to draw using some learning-based methods, followed by a defined cleaning sequence to clean the board itself to reset for its next iteration of experiment/trial.

4.3 Methodology

4.3.1 Picking Up and Placing Back the Sanitiser Bottle

To grab the sanitiser bottle upside down to pour the sanitiser 'fluid', We created a sequence in which the robot arm would first pick up the bottle with a grip parallel to the ground and dropping the bottle with a downward facing grip before picking it up again with a parallel grip and rotating to the downward grip, to get into a position for squeezing and dropping the fluid. To place the sanitiser bottle back right-side up, we reversed the motion that was used to pick it up upside down.

4.3.2 Squeezing 'Sanitiser' from the Bottle

To squeeze out sanitiser 'fluid', we added another gripping position that is just right to squeeze the 'fluid' out of the bottle without crushing it. The gripper will also revert to its non-squeeze position but a grip that is enough to hold the bottle.

4.3.3 Picking Up and Placing Back the Eraser

This is the same as transferring a cube in Task 2. An eraser was designed as in Appendix B.2.

4.3.4 Erasing Motion

To perform the erasing motion, we had to apply an appropriate amount of pressure on the drawing board while performing a sweeping motion. This is in a way similar to the drawing sequence in Task 3, although the motion is different and that the robot had to press harder on the drawing board. Appropriate adjustments from the drawing sequence was made to perform this task.

Appendix

A. Inverse Kinematics

$$\phi = \begin{cases} \frac{\pi}{2}, \text{if gripper facing up from ground} \\ \pi, \text{if gripper parallel to ground and facing inward from robot} \\ \frac{3\pi}{2}, \text{if gripper facing down to ground} \\ 0 \text{ or } 2\pi, \text{ if gripper parallel to ground and facing outward from robot} \end{cases} \tag{A.1}$$

$$\theta_1 = \tan^{-1}\left(\frac{y_4}{x_4}\right) \tag{A.2}$$

$$x_3 = x_4 - L_4 \cdot \cos(\theta_1) \cdot \cos(\phi) \tag{A.3}$$

$$y_3 = y_4 - L_4 \cdot \sin(\theta_1) \cdot \cos(\phi) \tag{A.4}$$

$$z_3 = z_4 - L_4 \cdot \sin(\phi) \tag{A.5}$$

$$D_{1,4}^2 = x_3^2 + y_3^2 + z_3^2 \tag{A.6}$$

$$D_{2,4}^2 = x_3^2 + y_3^2 + (z_3 - L_1)^2 \tag{A.7}$$

$$\theta_3 = 2\pi \ - \ \cos^{-1}\left(\frac{D_{2,4}^2 - L_2^2 - L_3^2}{2 \cdot L_2 \cdot L_3}\right) \tag{A.8}$$

$$\theta_2 = \begin{cases} \cos^{-1}\left(\frac{L_3^2 - L_2^2 - D_{2,4}^2}{2 \cdot L_2 \cdot L_3}\right) + \tan^{-1}\left(\frac{z_3 - L_1}{\sqrt{x_3^2 + y_3^2}}\right), D_{2,4} < 0 \\ \left[\pi \ - \cos^{-1}\left(\frac{L_3^2 - L_2^2 - D_{2,4}^2}{2 \cdot L_2 \cdot L_3}\right)\right] + \tan^{-1}\left(\frac{z_3 - L_1}{\sqrt{x_3^2 + y_3^2}}\right), \text{otherwise} \end{cases} \tag{A.9}$$

$$\theta_4 = \phi - \theta_2 - \theta_3 \tag{A.10}$$

B. CAD Model

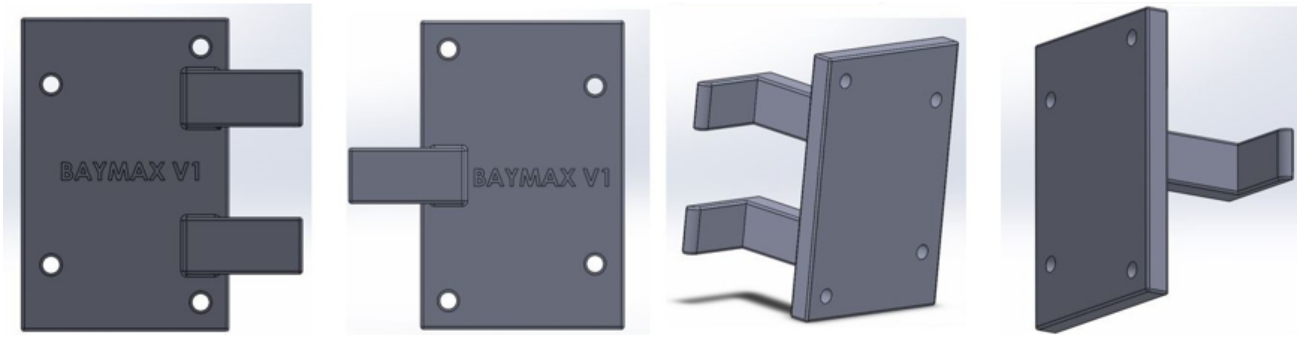

Figure B.1: CAD design parts of the pen gripper.

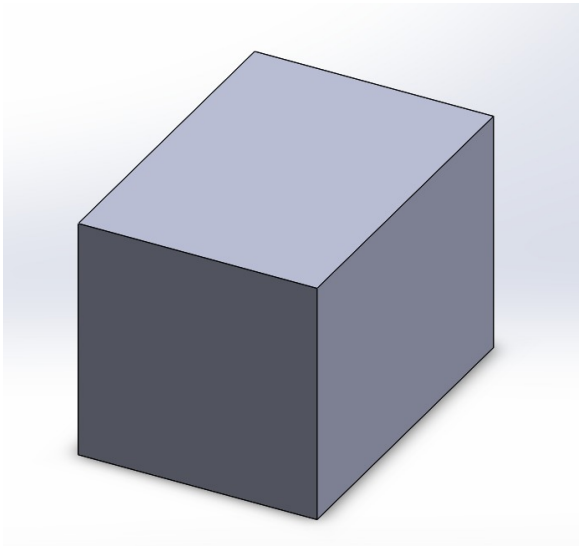

Figure B.2: CAD design of the eraser.

Bibliography

[1] ROBOTIS e-Manual. 2022. ROBOTIS e-Manual. [online]
Available at: https://emanual.robotis.com/docs/en/platform/openmanipulator_x/specification/dimension
[Accessed 15 February 2022].