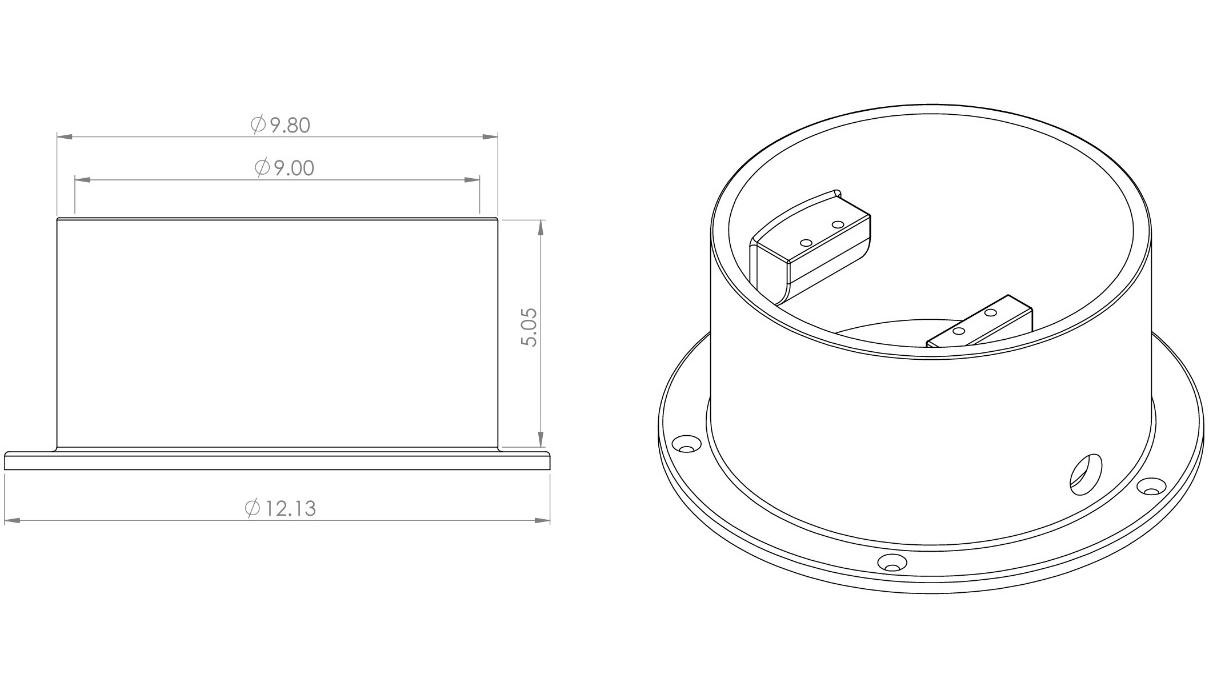
Faculty Signature

**AY 2024-25**

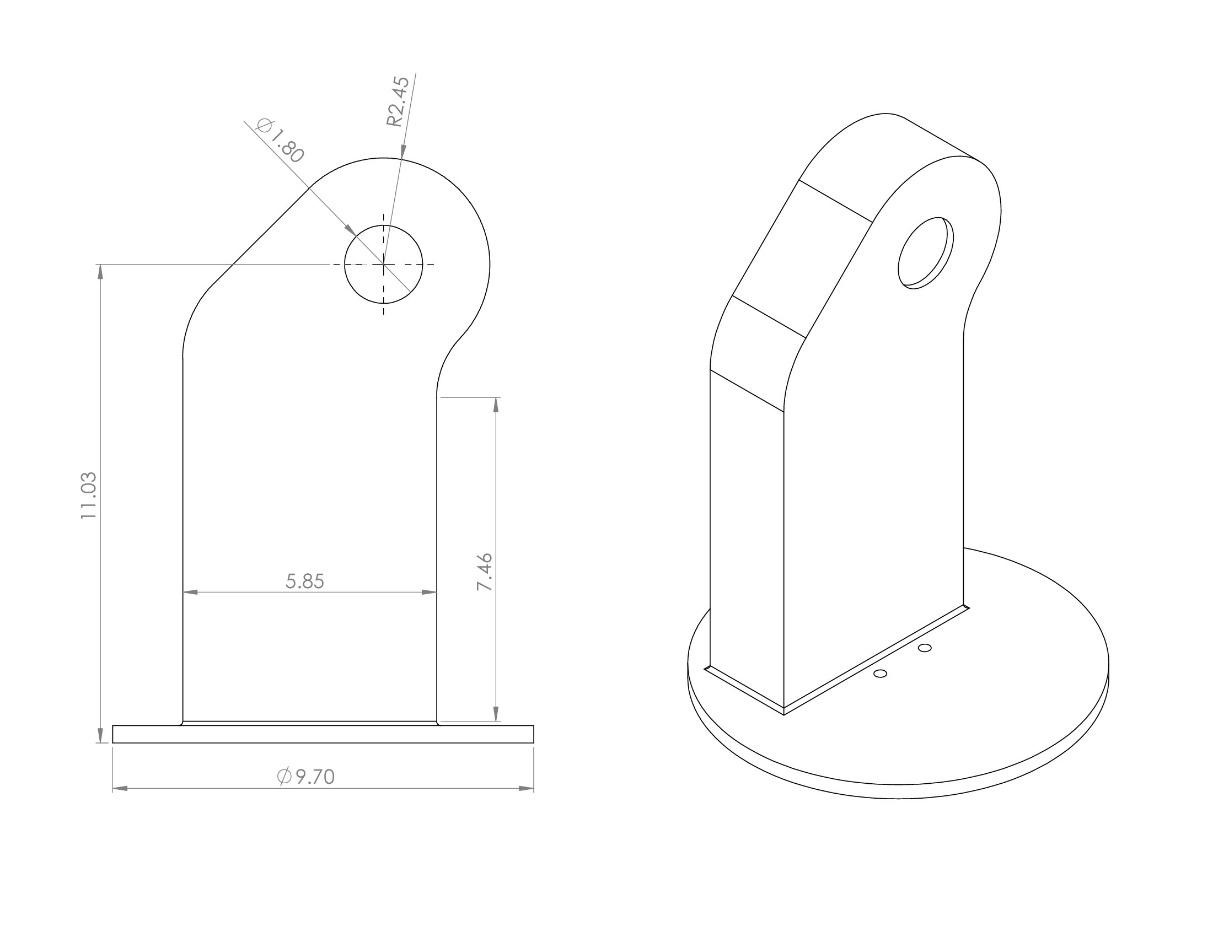
**DESIGN AND FABRICATION**

**1. Conceptual Design**

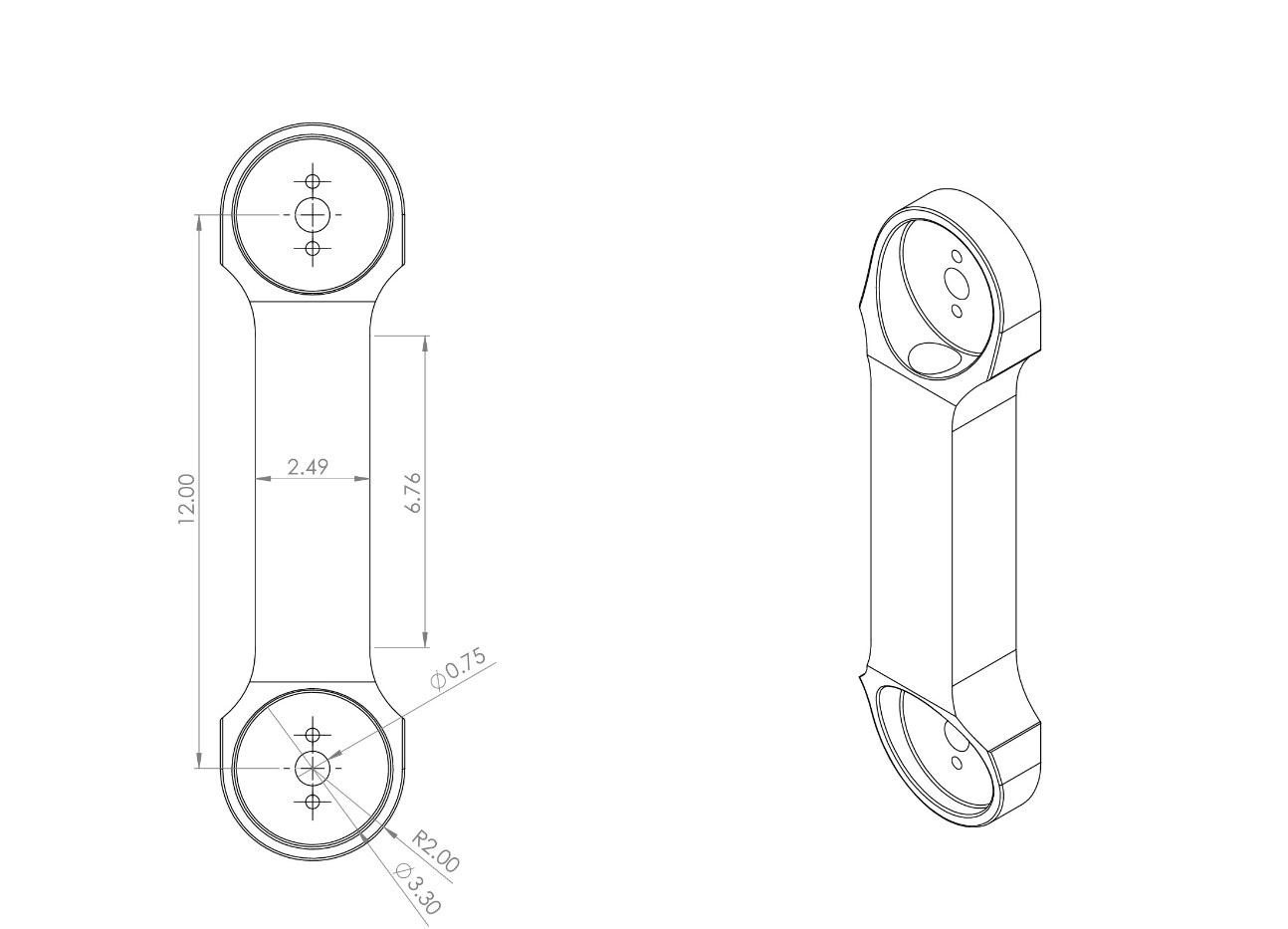
### 2D Model



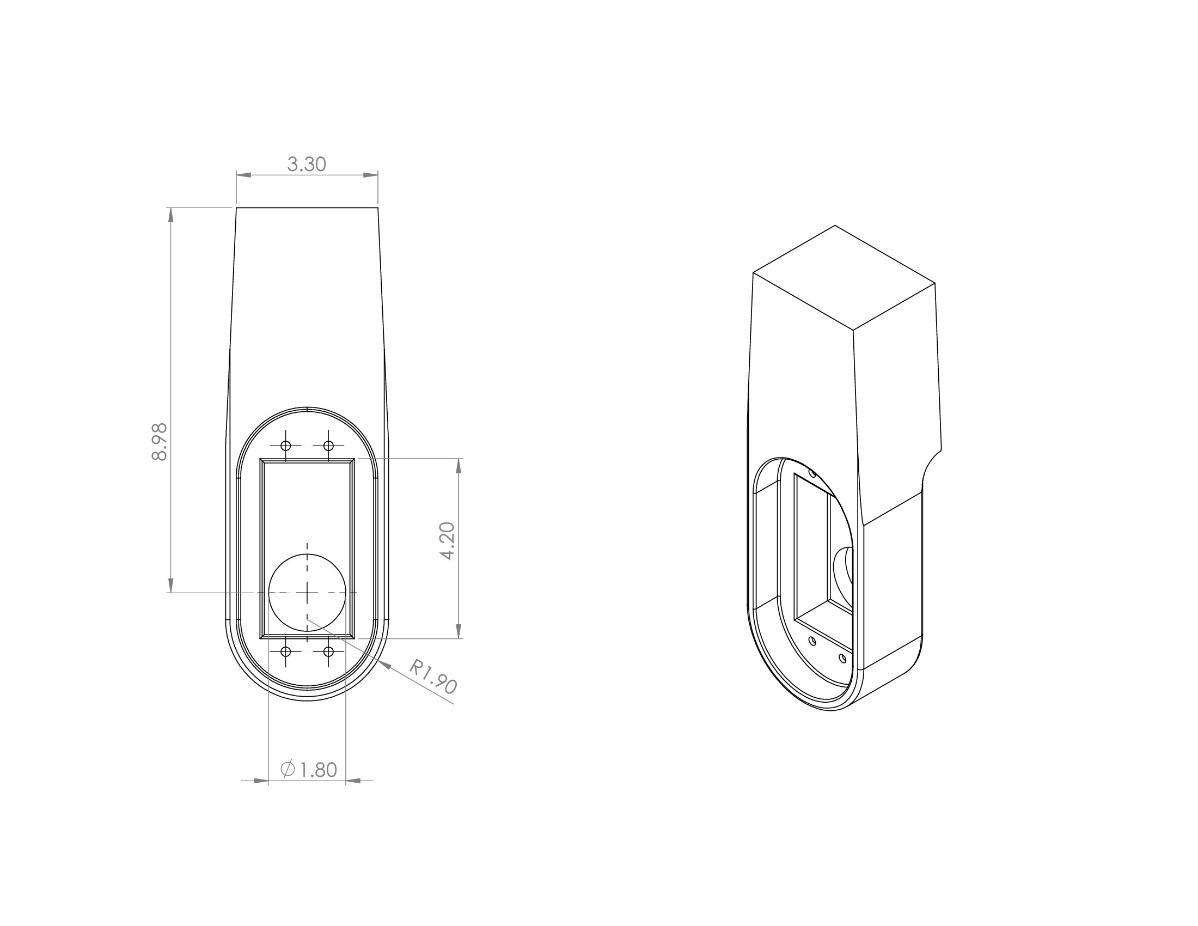
### Base



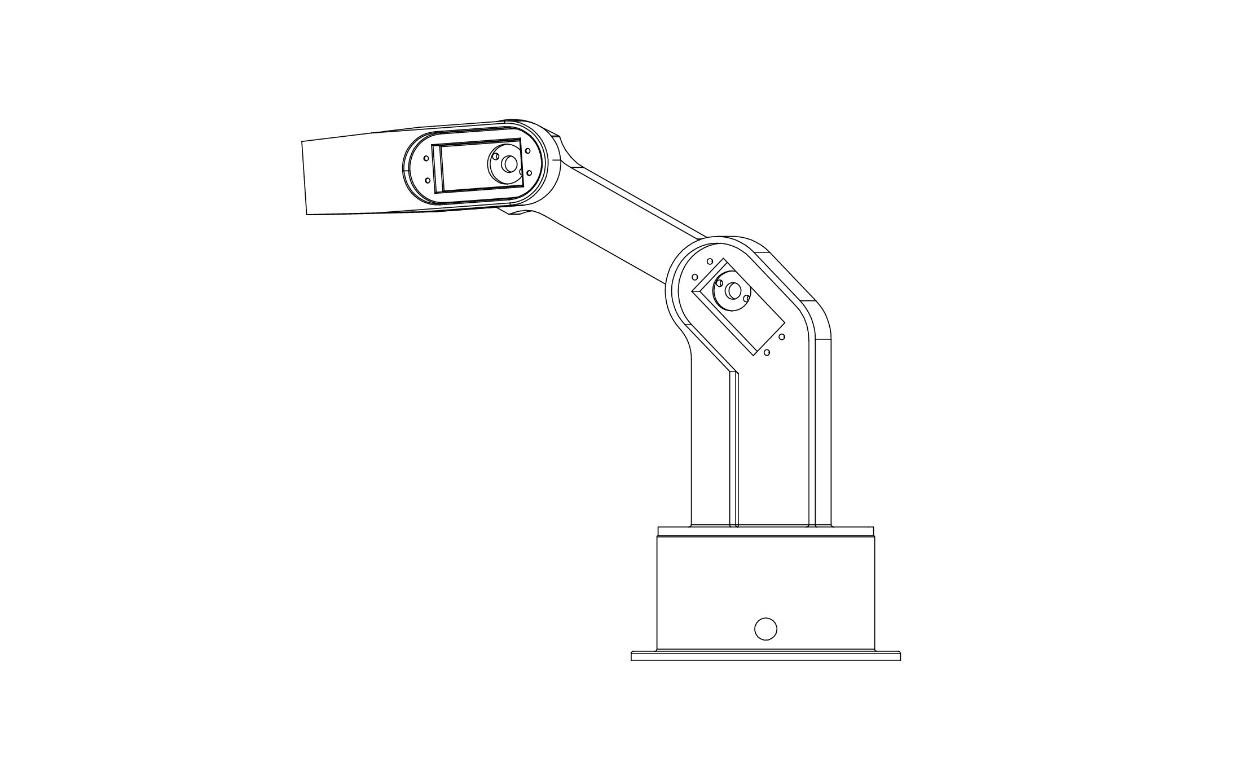
# Link 1



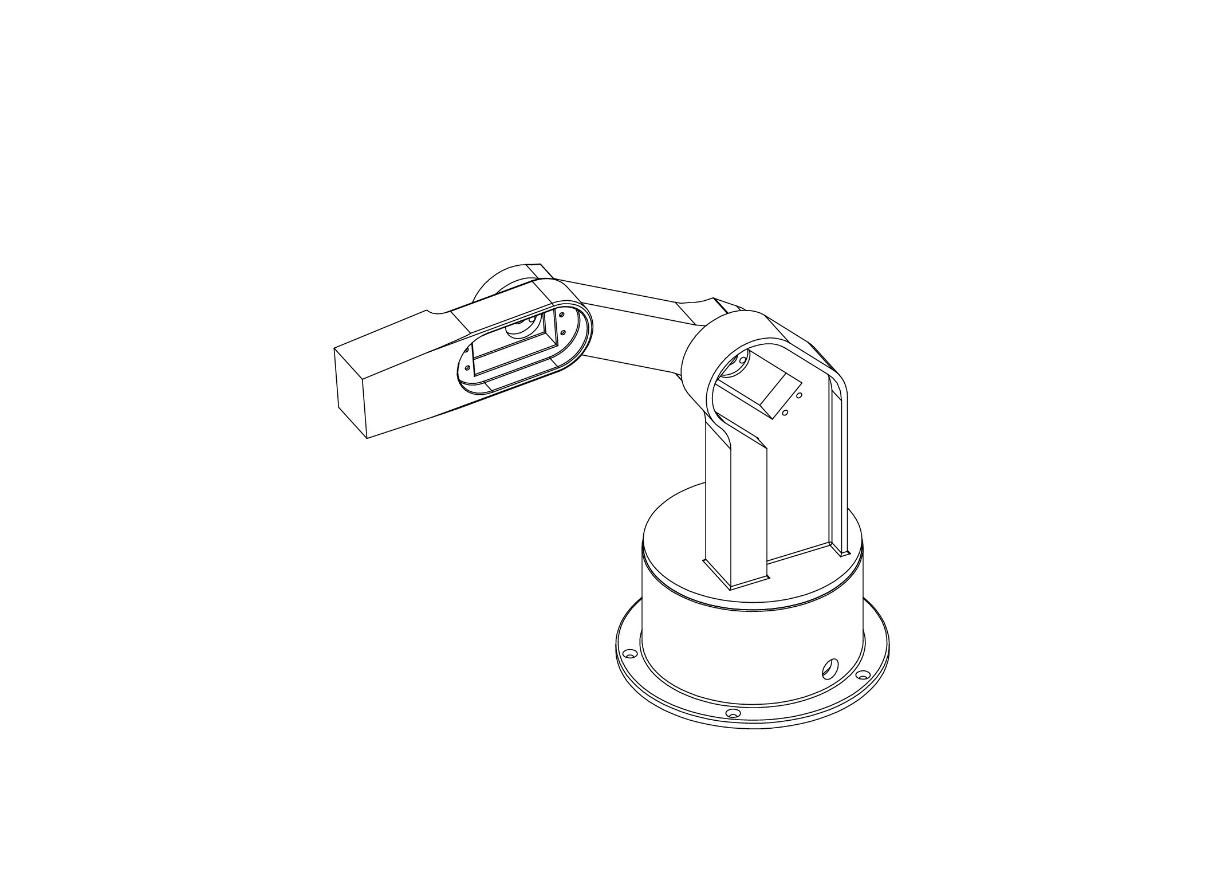
## Link 2



### Link 3

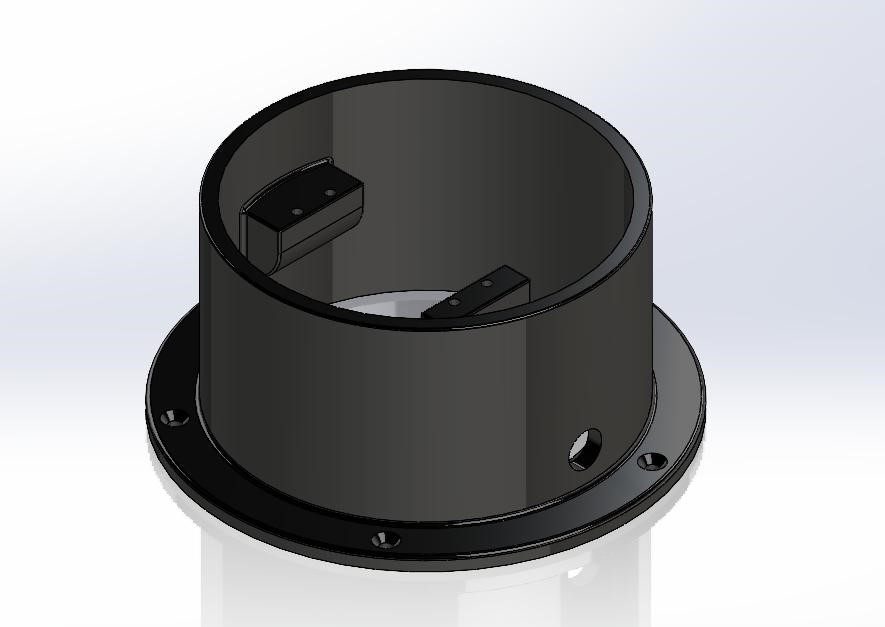


### Assembly (Side view)

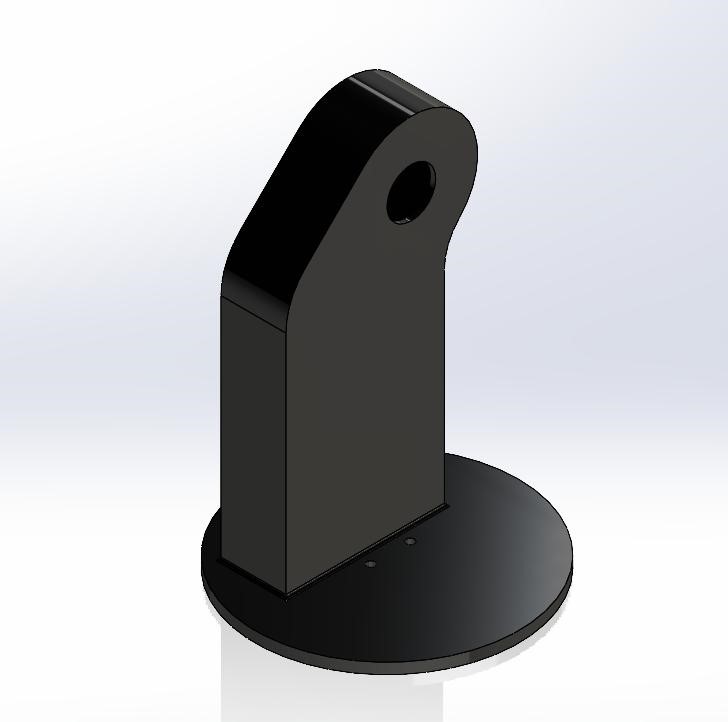


**Assembly (Isometric view)**

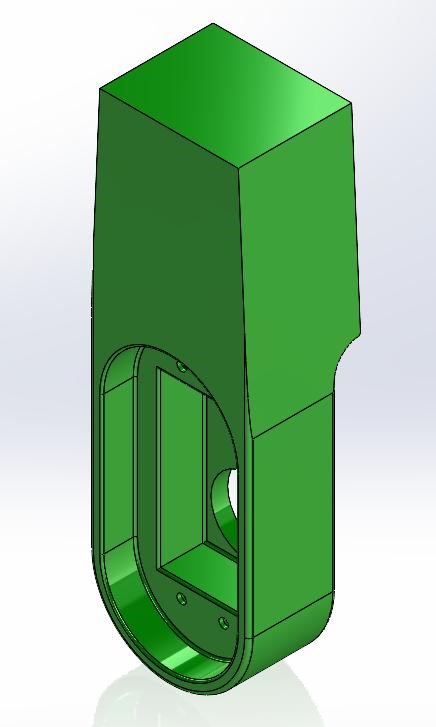
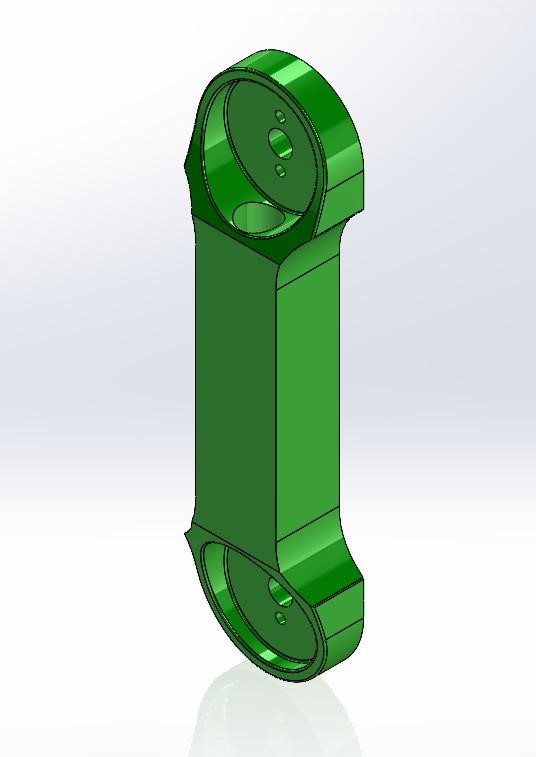
### 3D Model



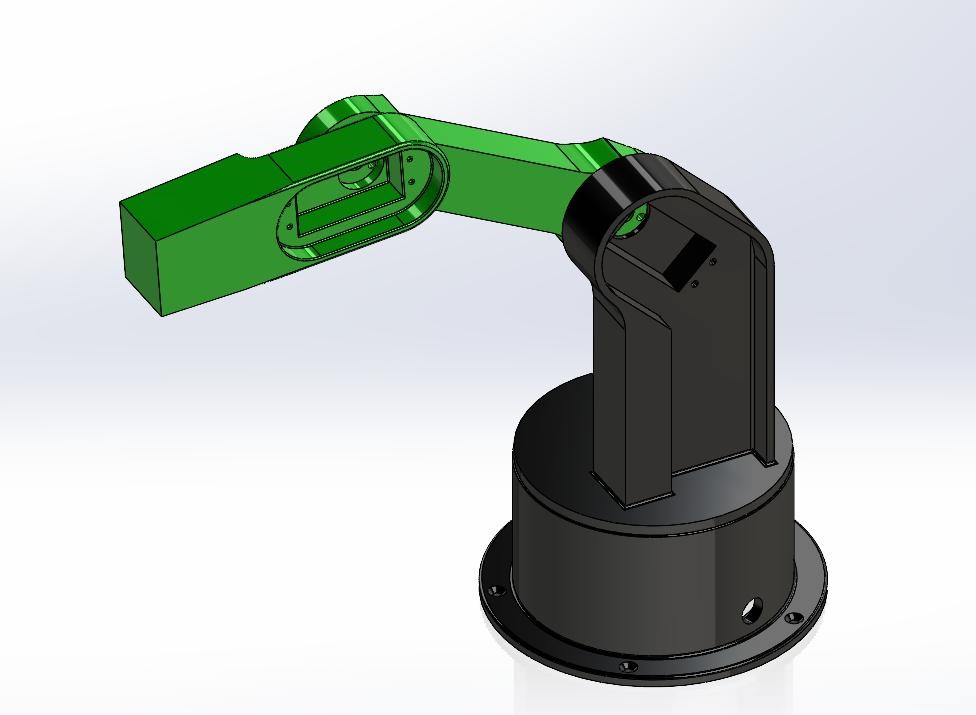
### Base



### Link 1



### Link 2 Link 3



**Assembly**

### 3D Printing



### Base Link 1

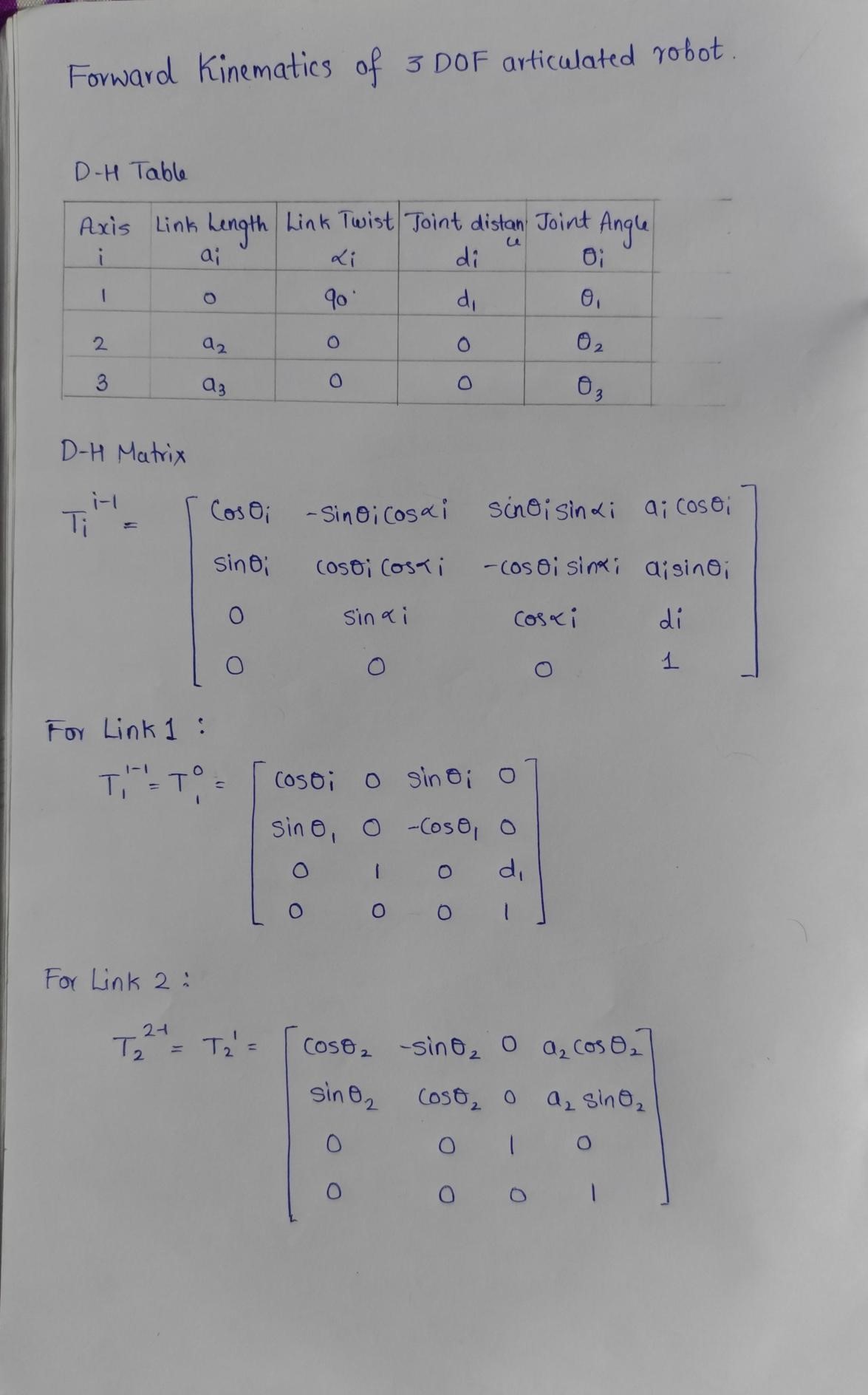


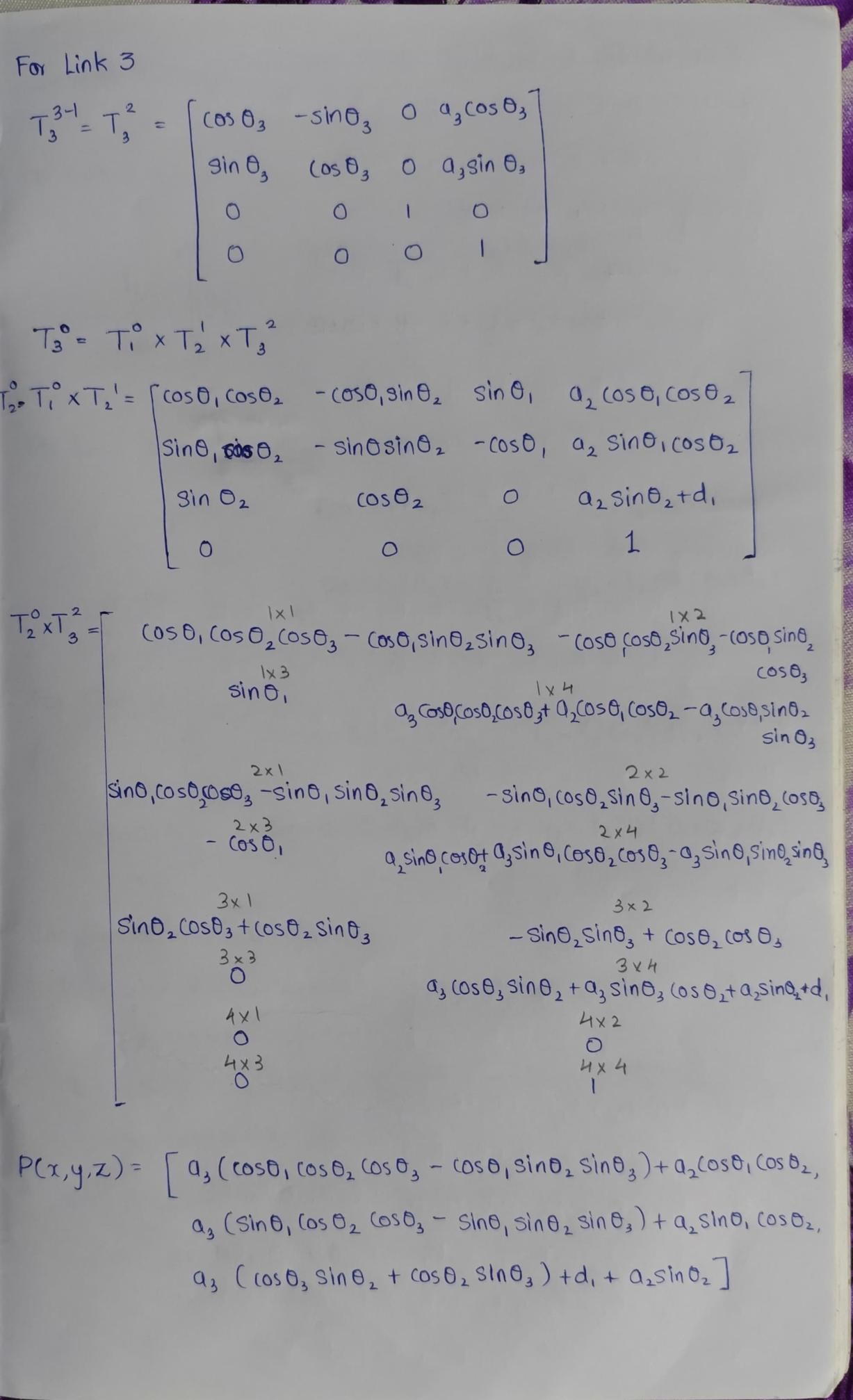
### Link 2 Link 3

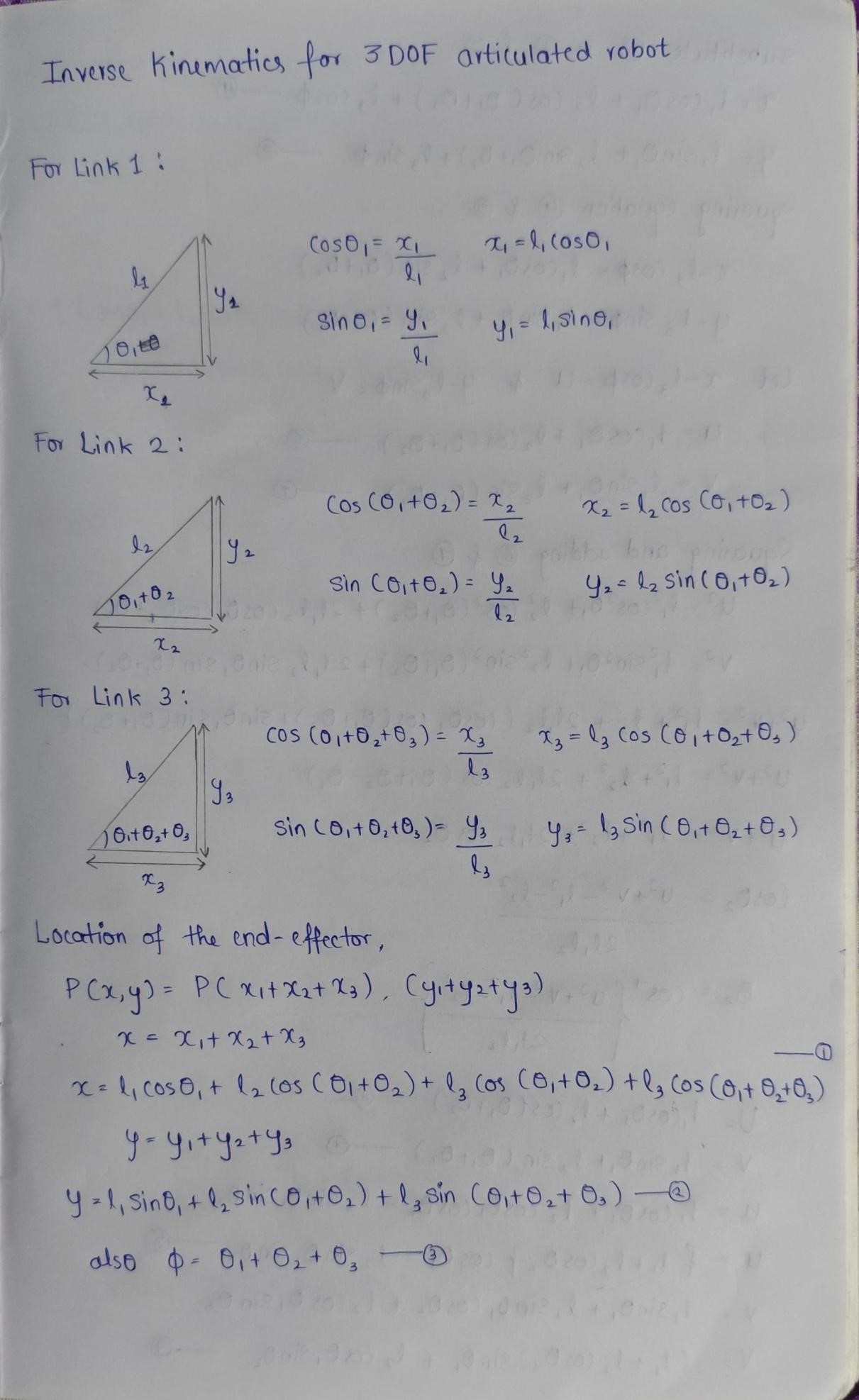


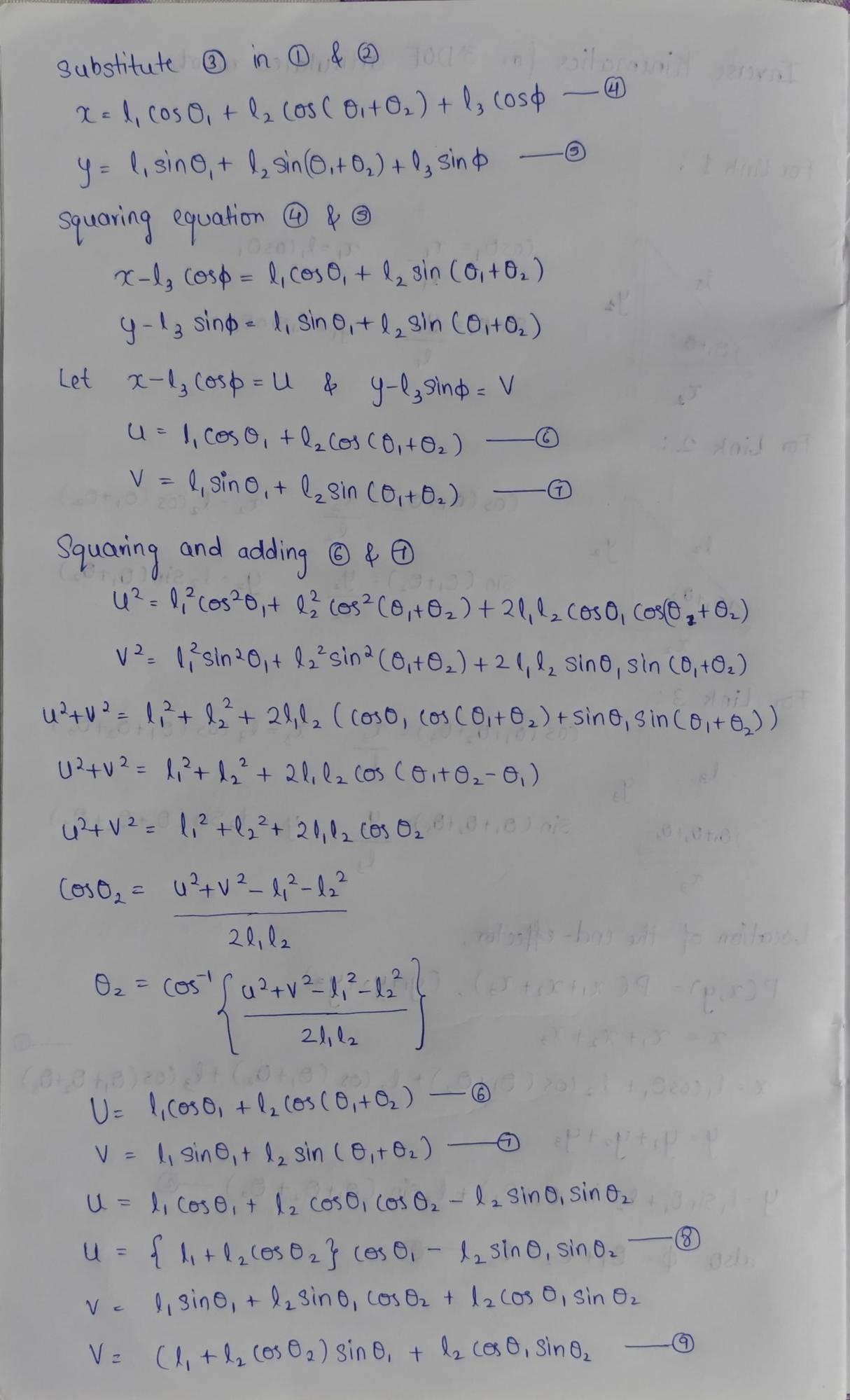
**Assembly**

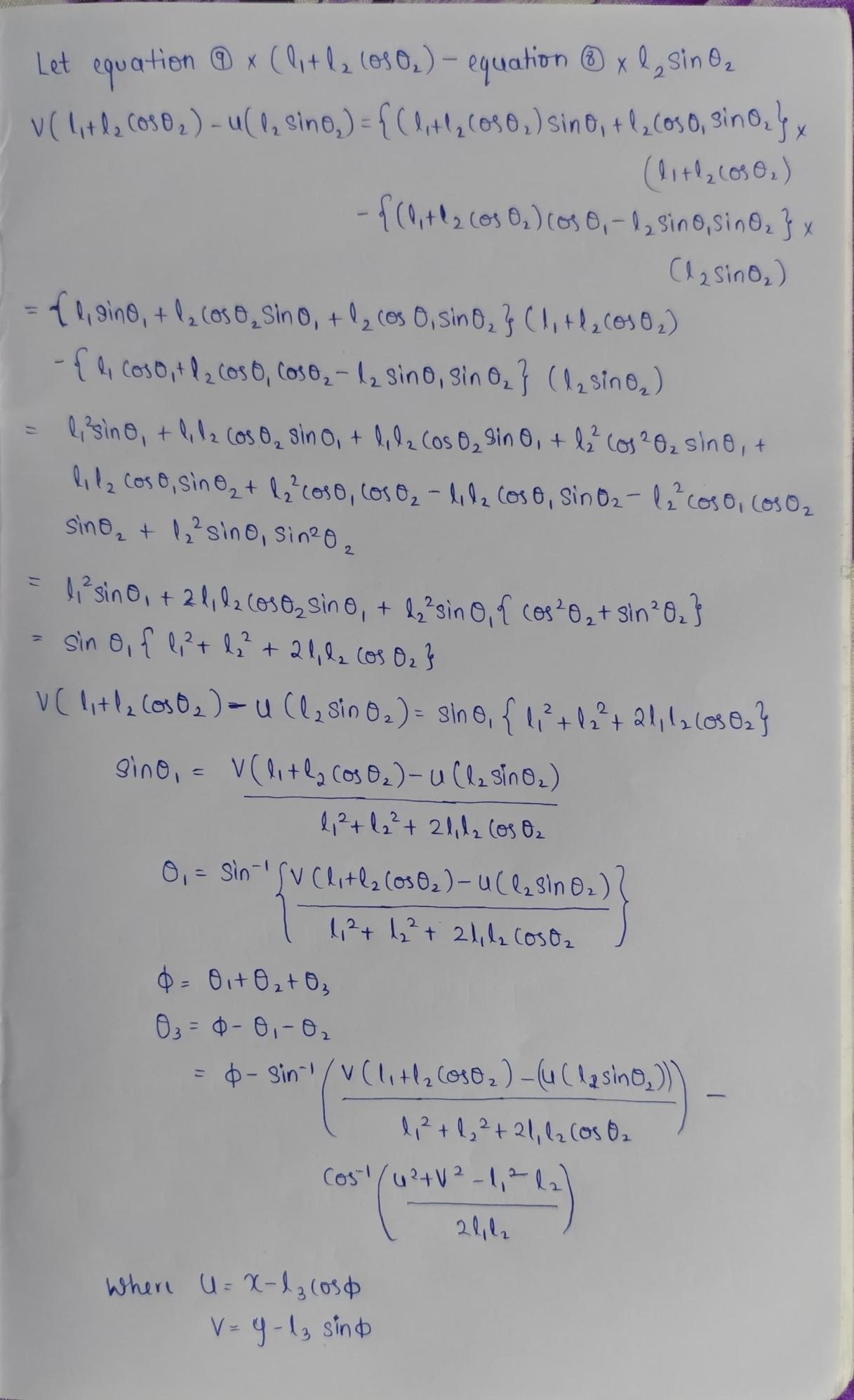
### 2. Forward and Inverse Kinematics solutions











### 3. Forward and Inverse kinematics codes

#include <Wire.h>

#include <Adafruit\_PWMServoDriver.h>

// Define PCA9685 object

Adafruit\_PWMServoDriver pwm = Adafruit\_PWMServoDriver();

// Arm segment lengths (in cm) const float L1 = 11.03; // Length of the first arm segment const float L2 = 12; // Length of the second arm segment const float L3 = 8.98; // Length of the third arm segment

// Servo range constants

#define BASE\_SERVO\_MIN 150

#define BASE\_SERVO\_MAX 600

#define SHOULDER\_SERVO\_MIN 150

#define SHOULDER\_SERVO\_MAX 600

#define ELBOW\_SERVO\_MIN 150 #define ELBOW\_SERVO\_MAX 600

void setup() {

// Start serial communication

Serial.begin(9600);

Serial.println("3-DOF Robotic Arm Control");

// Initialize PCA9685 driver

pwm.begin();

pwm.setPWMFreq(60); // Standard 60 Hz for servos delay(10);

}

void loop() {

if (Serial.available() > 0) {

char mode = Serial.read(); // Read mode (F for Forward, I for Inverse)

if (mode == 'F') {

// Forward Kinematics Mode

Serial.println("Enter Theta1 (Base servo), Theta2 (Shoulder servo), and Theta3 (Elbow servo), separated by space:");

while (Serial.available() < 3)

;

float theta1 = Serial.parseFloat(); // Base servo angle (0 to 90 degrees) float theta2 = Serial.parseFloat(); // Shoulder servo angle (0 to 180 degrees) float theta3 = Serial.parseFloat(); // Elbow servo angle (0 to 180 degrees)

// Convert to radians theta1 = radians(theta1); theta2 = radians(theta2); theta3 = radians(theta3);

// Calculate end effector position (X, Y, Z)

float x = L1 \* cos(theta1) + L2 \* cos(theta1 + theta2) + L3 \* cos(theta1 + theta2 + theta3);

float y = L1 \* sin(theta1) + L2 \* sin(theta1 + theta2) + L3 \* sin(theta1 + theta2 + theta3);

float z = L3 \* sin(theta3); // Assuming some vertical component contributed by L3

// Display calculated position

Serial.print("End Effector Position: X = ");

Serial.print(x);

Serial.print(", Y = ");

Serial.print(y);

Serial.print(", Z = ");

Serial.println(z);

// Map the angles to PWM values and set PWM for servos int basePWM = map(theta1 \* 180.0 / PI, 0, 90, BASE\_SERVO\_MIN,

BASE\_SERVO\_MAX);

int shoulderPWM = map(theta2 \* 180.0 / PI, 0, 180, SHOULDER\_SERVO\_MIN,

SHOULDER\_SERVO\_MAX);

int elbowPWM = map(theta3 \* 180.0 / PI, 0, 180, ELBOW\_SERVO\_MIN,

ELBOW\_SERVO\_MAX);

pwm.setPWM(0, 0, basePWM); // Move base servo pwm.setPWM(1, 0, shoulderPWM); // Move shoulder servo pwm.setPWM(2, 0, elbowPWM); // Move elbow servo

} else if (mode == 'I') {

// Inverse Kinematics Mode

Serial.println("Enter X, Y, and Z (in cm), separated by space:"); while (Serial.available() < 3)

;

float x = Serial.parseFloat(); // X position float y = Serial.parseFloat(); // Y position float z = Serial.parseFloat(); // Z position

// Inverse Kinematics for 3-DOF float r = sqrt(x \* x + y \* y); // Radial distance float d = sqrt(r \* r + z \* z); // Total distance from origin if (d > (L1 + L2 + L3)) {

Serial.println("Target out of reach!");

} else { // Base angle

float theta1 = atan2(y, x);

// Law of cosines for shoulder and elbow

float cosTheta2 = (d \* d - L1 \* L1 - L2 \* L2) / (2 \* L1 \* L2); float theta2 = acos(cosTheta2); // Shoulder angle float theta3 = atan2(z, r); // Elbow angle

// Convert to degrees theta1 = degrees(theta1); theta2 = degrees(theta2); theta3 = degrees(theta3); // Display joint angles

Serial.print("Theta1 = ");

Serial.print(theta1);

Serial.print(", Theta2 = ");

Serial.print(theta2);

Serial.print(", Theta3 = ");

Serial.println(theta3);

// Map the angles to PWM values and set PWM for servos

int basePWM = map(theta1, 0, 90, BASE\_SERVO\_MIN, BASE\_SERVO\_MAX); int shoulderPWM = map(theta2, 0, 180, SHOULDER\_SERVO\_MIN,

SHOULDER\_SERVO\_MAX);

int elbowPWM = map(theta3, 0, 180, ELBOW\_SERVO\_MIN,

ELBOW\_SERVO\_MAX);

pwm.setPWM(0, 0, basePWM); // Move base servo pwm.setPWM(1, 0, shoulderPWM); // Move shoulder servo pwm.setPWM(2, 0, elbowPWM); // Move elbow servo

}

}

}

}

### 4. Results

