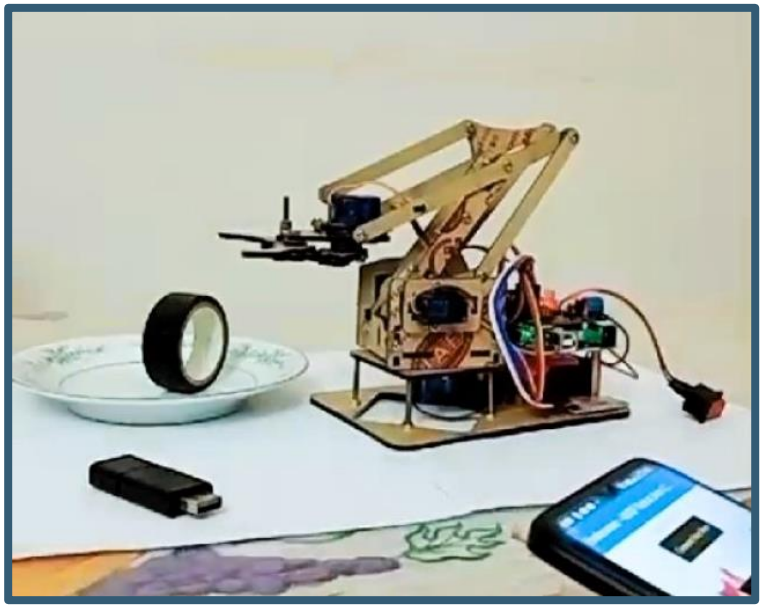
Revision 3 – Research Proposal

Endeavour - 4DOF Robot Arm

Small-Scale Experimentation and Prototype Development

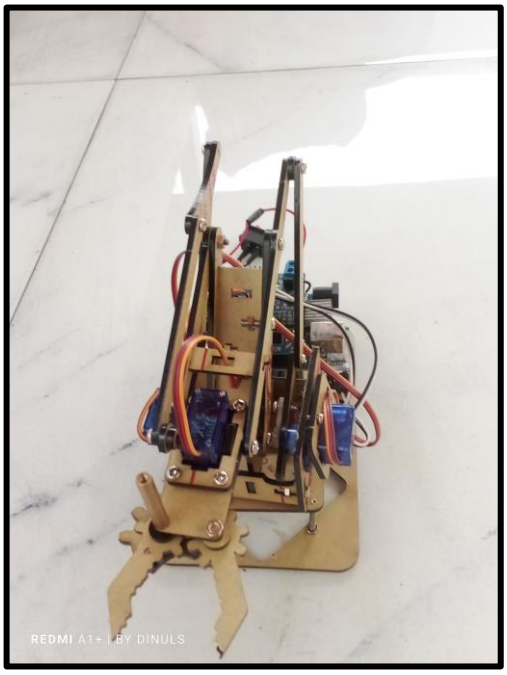


W.V.Dinul Sasnada Imandiv
Chanul Hansana
Manitha Ishen Waduge
ENDEAVOUR

Small-Scale Experimentation and Prototype Development

Endeavour - 4DOF Robot Arm

W.V.Dinul Sasnada Imandiv
Chanul Hansana
Manitha Ishen Waduge



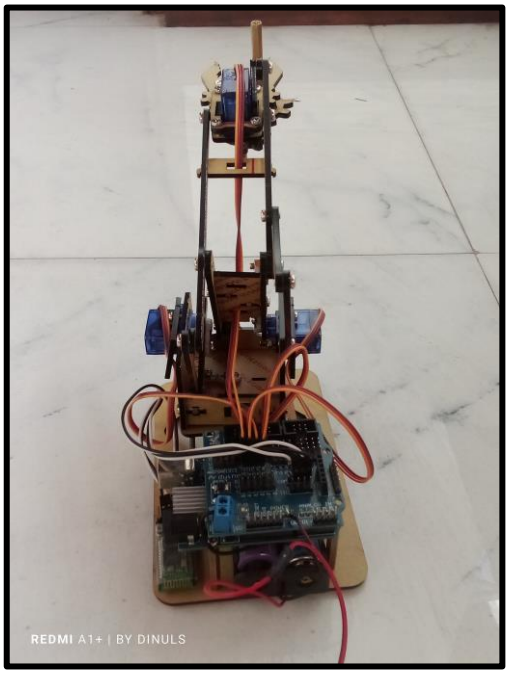


Figure 1.1 ,1.2: Front view of the arm (left), Back view of the arm (right) respectively.

1. Introduction

- The Endeavour - 4DOF Robot Arm is a small-scale robotic arm designed to demonstrate precise control using affordable, off-the-shelf components. It aims to perform basic tasks while integrating sensor-based interaction with its environment. Controlled via an Android app and Bluetooth, the project showcases how low-cost hardware can be used to build functional robotic systems.
- The Endeavour project focuses on designing a 4 Degree of Freedom (4DOF) robotic arm using commercially available components like Tower Pro SG90 Servos and an Arduino Uno microcontroller, housed in an acrylic chassis. The arm, while limited in lifting capacity, provides a functional platform for small-scale experimentation, allowing further research in robotic control systems.

2. Research Objectives & Targets

"Our target is to make an affordable, simple, free and open-source test-bed for robot arm experimentation."

W.V.Dinul Sasnada Imandiv

1. Primary Objectives:

- Prototype Development: Design and assemble a 4DOF robotic arm with a lifting capacity of 55 grams using an acrylic chassis and Tower Pro SG90 Servos.
- Safety Mechanisms: Incorporate features like a kill switch and a hardcoded Servo movement range limits for arm's longevity and flawless functioning.
- Bluetooth-Based Control: Implement an Android app to control the robot in real-time via Bluetooth (HC-05).

2. Secondary Objectives:

- Optimize Servo Control: Develop algorithms that ensure smooth, precise movement, reducing jitter and vibrations.
- Stability Testing: Evaluate the structural stability under weight handling constraints and refine as needed.

3. Research Problems

1. Core Problem:

- Designing a low-cost robotic arm capable of smooth, accurate movements under weight constraints and real-time interaction with the environment.

2. Specific Challenges:

- Servo Performance: Ensuring stability and reducing jitter and vibration with Tower Pro SG90 Servos.
- Chassis Limitations: Acrylic material limits weight capacity to 55 grams, making structural integrity a concern.

4. Methodology

1. Design and Prototype Development:

- Mechanical Assembly: Build the robot using an acrylic chassis, and SG90 Servos for movement.
- Arduino Setup: Use Arduino Uno and a Sensor Shield to control Servos and sensors, optimizing power distribution and connections, plus the basic control algorithms to move joints within a range of angles in 5-degree increments (10-degrees for Base Servo).

Servo Name	Connected Pin no.	Idle Position (in degrees)	Minimum Position	Maximum Position
Base	8	90	0	180
Shoulder	9	90	0	180
Elbow	10	90	60	180
Gripper	11	90	0	180

a. Special Code Features:

1. Servo Control Setup:

- The code initializes servos for the base, shoulder, elbow, and wrist.
- Each servo has position limits ('BASE_MIN', 'BASE_MAX', etc.), with specific ranges for safety.

```
1. const int BASE_MIN = 0, BASE_MAX = 180;
2. const int SHOULDER_MIN = 60, SHOULDER_MAX = 180;
3. const int ELBOW_MIN = 0, ELBOW_MAX = 180;
4. const int WRIST_MIN = 0, WRIST_MAX = 180;
```

2. Smooth Movement Implementation:

```
 1. bool isMovingBase = false, isMovingShoulder = false, isMovingElbow = false,
isMovingWrist = false;
 2.
 3. void moveServoSmoothly(Servo &servo, int currentPos, int targetPos, int step = 1) {
 4.   targetPos = constrain(targetPos, 0, 180);
 5.   if (currentPos != targetPos) {
 6.     if (abs(currentPos - targetPos) <= step) {
 7.       currentPos = targetPos; // Snap to target
 8.     } else {
 9.       currentPos += (currentPos < targetPos) ? step : -step;
10.     }
11.     servo.write(currentPos);
12.   }
13. }
14.
```

- A ‘moveServoSmoothly’ function ensures each servo moves smoothly to its target position by adjusting the position gradually, avoiding abrupt jumps.
- This function also includes a "snap" feature that aligns the servo directly with the target when it's close, maintaining precision.

3. Individual Movement Functions:

- Separate functions (e.g., ‘moveBase’, ‘moveShoulder’) constrain each joint’s movement within set ranges, enhancing safety and preventing over-extension.

```
1. void moveBase(int delta) {
2.   targetBasePos = constrain(basePos + delta, BASE_MIN, BASE_MAX);
3.   isMovingBase = true;
4. }
5.
6. void moveShoulder(int delta) {
7.   targetShoulderPos = constrain(shoulderPos + delta, SHOULDER_MIN, SHOULDER_MAX);
8.   isMovingShoulder = true;
9. }
10.
11. void moveElbow(int delta) {
12.   targetElbowPos = constrain(elbowPos + delta, ELBOW_MIN, ELBOW_MAX);
13.   isMovingElbow = true;
14. }
15.
16. void moveWrist(int delta) {
17.   targetWristPos = constrain(wristPos + delta, WRIST_MIN, WRIST_MAX);
18.   isMovingWrist = true;
19. }
20.
```

4. Timing Control:

- A timing variable (‘previousMillis’) and interval (‘updateInterval’) manage the update rate, ensuring consistent movement without overwhelming the servos.

```
1. unsigned long currentMillis = millis();
2.   if (currentMillis - previousMillis >= updateInterval) {
3.     previousMillis = currentMillis; // Update the last update time
4.
5.     // Move servos smoothly
6.     if (isMovingBase) {
7.       moveServoSmoothly(baseServo, basePos, targetBasePos);
8.       if (basePos == targetBasePos) isMovingBase = false;
9.     }
10.    if (isMovingShoulder) {
11.      moveServoSmoothly(shoulderServo, shoulderPos, targetShoulderPos);
12.      if (shoulderPos == targetShoulderPos) isMovingShoulder = false;
13.    }
14.    if (isMovingElbow) {
15.      moveServoSmoothly(elbowServo, elbowPos, targetElbowPos);
16.      if (elbowPos == targetElbowPos) isMovingElbow = false;
17.    }
18.    if (isMovingWrist) {
19.      moveServoSmoothly(wristServo, wristPos, targetWristPos);
20.      if (wristPos == targetWristPos) isMovingWrist = false;
21.    }
22.  }
23. }
24.
```

2. Experimentation and Testing:

- Weight Handling: Test the robot's ability to manipulate objects within the 55g payload limit.
- Functionality testing: Test the arm's functionality in various ranges of angles & battery loads.

- Improvements: Implement smooth Servo movements and usage of Millis() instead of delay() in codes for more efficiency.

3. Control System Implementation:

- Bluetooth Communication: Use an HC-05 Bluetooth module for real-time control via an Android app built with MIT App Inventor.
- Android App: Design the app with options for pre-set positions, fine-tuning controls, range controls, detach and re-attach buttons (kill switches) and auto-return to a safe position.

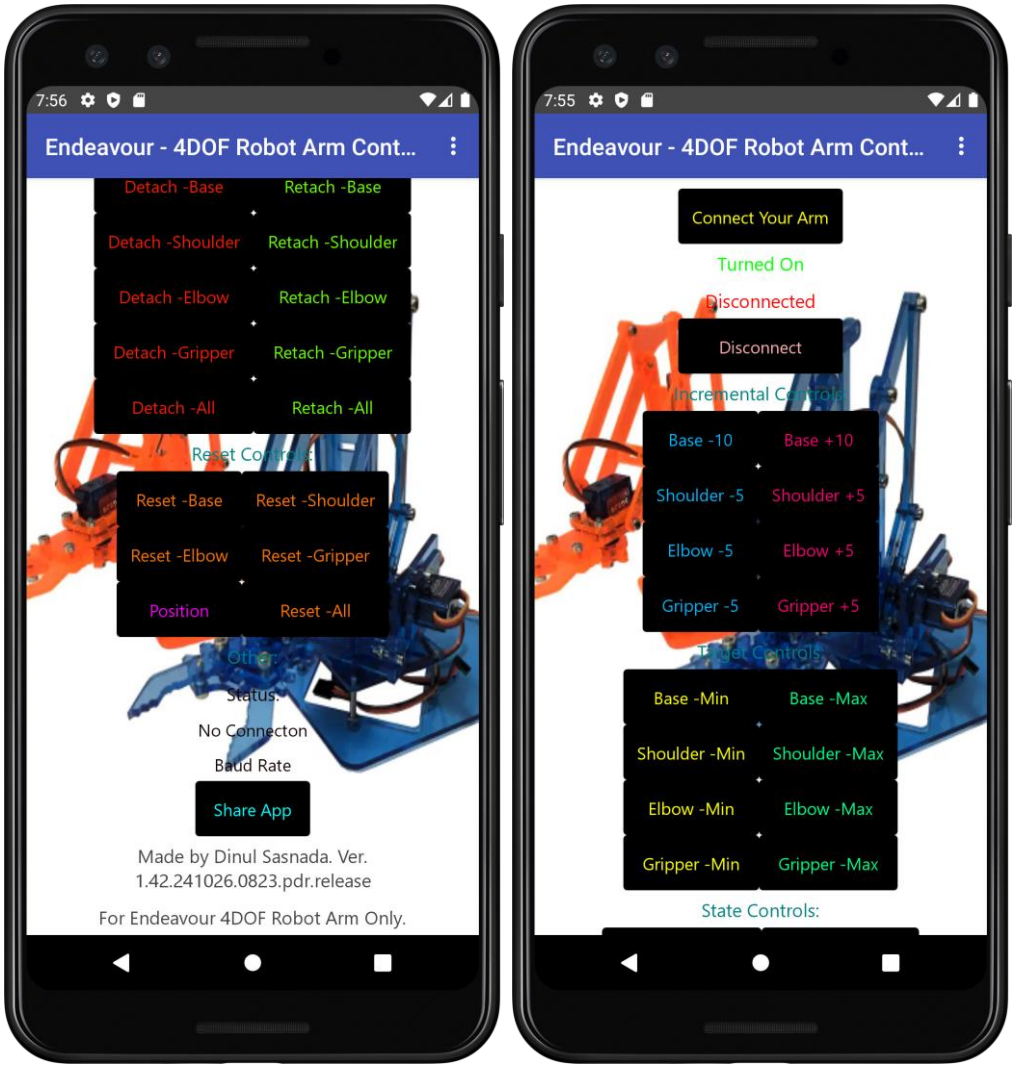


Figure 4.1 ,4.2: The "Endeavour – 4DOF Robot Arm Controller" running on a Google Pixel 3a (API Level 30) emulated in Android Studio, startup screen (left), bottom (right).

5. Experimentation and Expected Outcomes

1. Targets and Experiments:

- Object Handling: The arm was tested with small objects weighing up to 55 grams, evaluating its stability and precision.
- Performance: Conduct small-scale experiments to validate the accuracy and stability of the robotic arm.
- Safety: Test the functioning of the kill switch to ensure the safety of the arm.
- Control System: Bluetooth-based Android App with ability to command the arm to move in increments of 5 degrees (10 for Base) and immobilize the arm.
- Optimization: Implement a smooth Servo movement for reduced power usage and longevity of the arm.

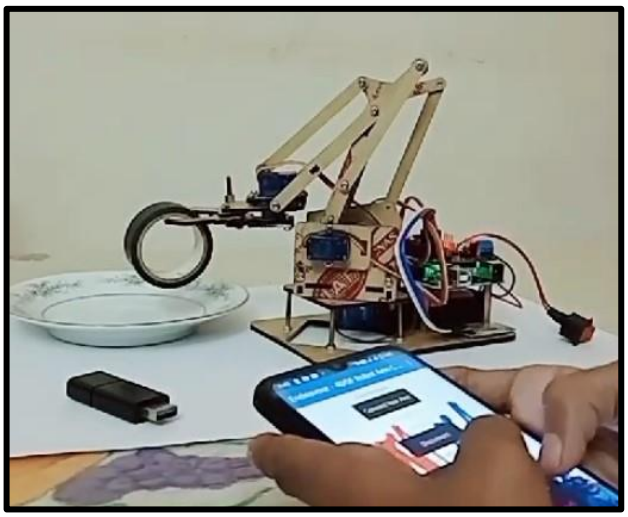


Figure 5.1: Assessing the arm's object handling capabilities and the control system's functionality.

2. Expected Outcomes

- A functional prototype of the 4DOF robotic arm, housed in a sturdy acrylic chassis and capable of performing basic manipulative tasks within a 50-gram weight limit.
- An Android app that can control and monitor the robotic arm in real-time via the HC-05 Bluetooth module.
- Implementation of a kill switch to immobilize the arm.

6. Timeline and Resources

1. Timeline:

- The project commenced on September 23rd and is ongoing. The primary focus has been on small-scale experimentation and prototype development to meet the research objectives.

23 Sep. 2024	24 Sep. 2024	27 Sep. 2024	30 Sep. 2024	06 Oct. 2024	15 Oct. 2024	18 Oct. 2024	20 Oct. 2024	24 Oct. 2024	25 Oct. 2024	26 Oct. 2024	31 Oct. 2024
First prototype was built.											
		First revision of the Android app was built with basic movement control.									
				Reveal-ed to public for first time.							
					The code and the app were optimized with more functions and better reliability.						
							Resolved all flaws and problems related to the functionality. Prepared for public performance.				
									First live perfor mance was done.		
										More improvements are done according to the public feedback and reviews.	

2. Resources:

- Hardware: Arduino Uno, Arduino Sensor Shield, Tower Pro SG90 servos, acrylic chassis, HC-05 Bluetooth module.

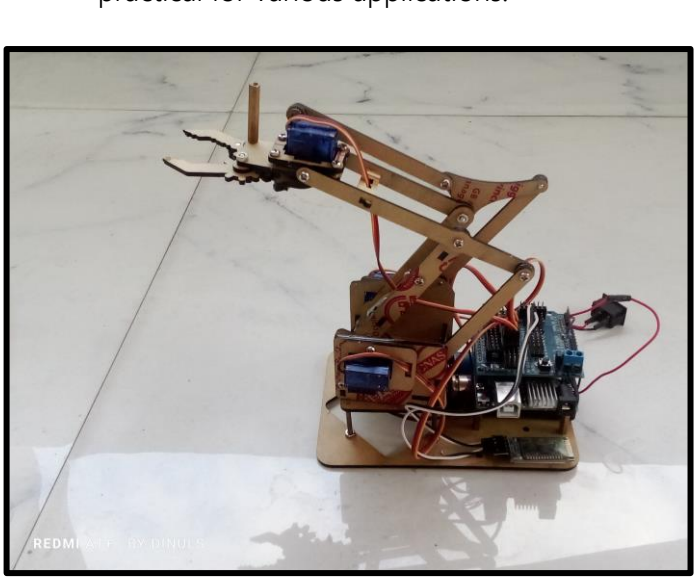
- Software: Avrdude, Arduino IDE, MIT App Inventor for the Android app, and simulation tools for kinematics.
- General Tools for assembling the chassis.

7. Potential Impact and Applications

- The Endeavour - 4DOF Robot Arm project can serve as a foundational prototype for further research in robotic manipulator development.
- The results and insights gained from this project can be applied to small-scale automation, educational purposes, and research on affordable robotic systems.

8. Conclusion

- The Endeavour - 4DOF Robot Arm project successfully achieved its objectives of building a low-cost robotic arm with real-time Bluetooth control, basic environmental awareness, and safety mechanisms.
- Future improvements in material strength, control algorithms, and sensor systems can further enhance its capabilities.
- Given the constraints of material strength and grip, the robotic arm is designed to lift a maximum of 55 grams. With the potential for significant impact in the field of low-cost robotic systems, this project aims to contribute to making robotics more accessible and practical for various applications.



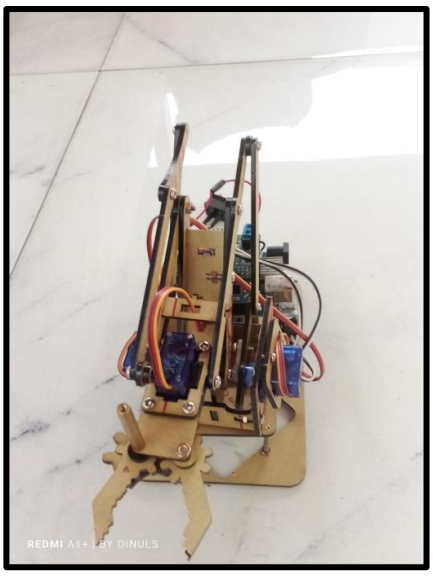


Figure 8.1, 8.2: Endeavour at the idle position (right, former spec), (left, new spec) respectively.