Robot Arm Project

Levi Vincent

Abd El Rahman Sidky

Tommy Jackson

December 2nd, 2025

1. Introduction

For this project we used a 3D printed robot arm with 5 degrees of freedom designed by Dejan. The robot consisted of several links each rotating on a revolute joint and the end effector was a claw. The goal for this project was to use this arm to pick up a rubber duck from a set point then drop it in a container. The duck's position was known. In order to compare our code to others', we participated in a competition. The competition tested how many ducks each group could grab and drop in a minute, manually replacing the duck for each cycle.

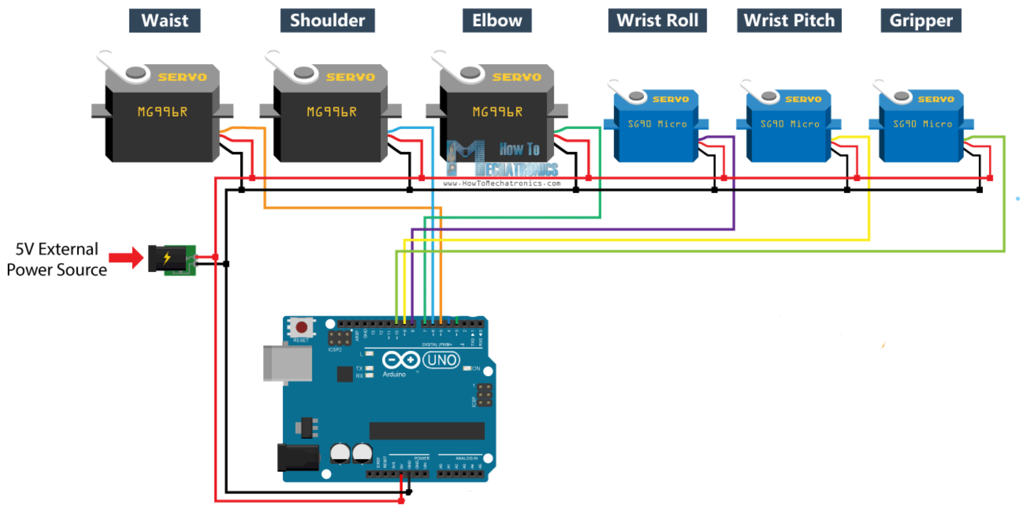
2. Materials

- 3D Printed Robot Arm (Dejan's Design)
- 5v, 2A, Power supply
- 3 SG90 9g Micro Servos 000
- 3 MG996R 55g Servos
- 1 Breadboard
- #4 Button-Head Bolts as Needed
- #4 Nuts as Needed
- Small Washers as Needed
- Jumper Wires as Needed
- 1 Rubber Band

3. Methods

To develop the system we first needed to create the physical arm. We were provided with the printed parts from Dejan's STL files. We removed the supports on all of these parts before assembling. We then assembled the arm with the pictures in Dejan's tutorial. Each piece was effectively connected by the servo on the previous piece. For the servo connections we screwed into the 3D printed material directly. If screws were too long to fit, we added washers and spacers to compensate. The claw assembly was assembled by making the general shape and adding a #4 bolt and nut at each hole. For the driven gear we added 3 washers below so it would interlock with the driving gear. We used fully threaded bolts since we had access to some, but for this project partially threaded would be better.

The circuit shown below was used. Each servo was connected to the voltage from the 5 Volt source. Each servo was then also connected to an arduino pin for the output. Since we only used 5 of the 6 servos for our path, the provided code uses pins 3-7. We used the external power source to ensure each servo was working, since the arduino output is not truly 5 Volts. On the robot, we used adhesives and zip ties to keep wires away from the claw.



In the beginning we were using Arduino IDE with the servo.h library to control the servos, but we had a lot of problems from stability to angles, so we developed our own code from scratch in AVR C. Instead of using a servo library, we developed our own code to control the servos using the internal AVR timer one. We used CTC mode with timer one to generate PWM with a 20 micro second period to control the servos. In the code we control the time of the high pulse which represents an angle.

In other words, a servo receives a PWM with a period of 20 micro seconds most of which is low pulse. While the time that signal is high is from 600 micro seconds to 2400 micro seconds, this spam presents the angle from 0 to 180 degrees.

Using this logic we control the time value between 600 micro seconds to 2400 micro for every servo to set its position. Then we created the pattern of movements for each servo through trial and error, a set of values to go through with delays in between to pick up the duck and place the duck and repeat.

4. Results and Discussion

Our code successfully picked up a duck from our selected square and dropped it behind itself. We found that placing the duck facing sideways (parallel to box), was best for this position to be picked up. We roughly estimated each duck to take ten seconds, so we estimated that we would manage to pick up six ducks during the minute.

In the competition we placed fifth. We managed to pick up five ducks, which is slightly less than our estimate. This was a fairly average score compared to the rest of the groups. During our run, we dropped or failed to pick up a few ducks, and had a slightly delayed start to the code uploading. We believe that we could have successfully transported eight ducks if the robot arm had performed optimally, which is more than our prediction. A score of eight would have been third place. We did successfully make the robot complete its task, and were not in a bad position compared to others, but we could still optimize the design some.

In comparison to the leaderboard groups we noticed a few things we could fix. First of all, we chose to grab the body of the duck, but we noticed that most consistent groups grabbed the head, so this may be the better choice for holding it. Another issue may have been our path, we chose to lift the duck, then turn the base, and drop. One successful strategy was to simply keep the base in the same spot and use the shoulder and elbow servos to flip behind itself. The other high scoring team simply positioned the duck closer, and thus had less distance to travel. Another thing we could alter in the future is increasing the weight on our plywood base, during the competition, it shifted slightly which may be why we failed to pick up a few ducks. One last general change would be to use lock nuts instead of normal nuts. Lock nuts help prevent self loosening when in operation, which is an issue we had when testing.

5. Conclusion

Overall, this project allowed us to design, program, and test a functional 5 DOF robot arm capable of repeatedly picking up and placing a rubber duck. By moving away from the arduino servo library and instead implementing our own PWM control using AVR C and timer one, we gained a deeper understanding of how different aspects of robot design each play an important role in the final product, such as understanding the relation between time and movement. Our robot was able to complete the main task and did fairly well in the competition, even though we only picked up 5 ducks instead of the 6 we had hoped for or the 8 a perfect run could have done.

6. Github Link

https://github.com/AbdelrahmanSidky/Robot-Arm-Project

7. Demo Video

https://www.youtube.com/shorts/6-eUyLpylPA

Or see github.

8. Acknowledgements

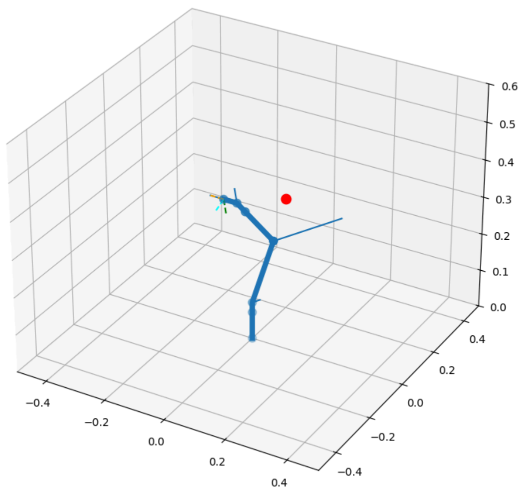
We would like to thank our professor and lab assistants for their guidance, feedback, and support throughout this project. We also want to thank Dejan for providing the 3D printed robot arm design that made this project possible. Finally, we would like to thank our classmates for their help during testing and for creating a fun and competitive environment during the duck grabbing challenge.

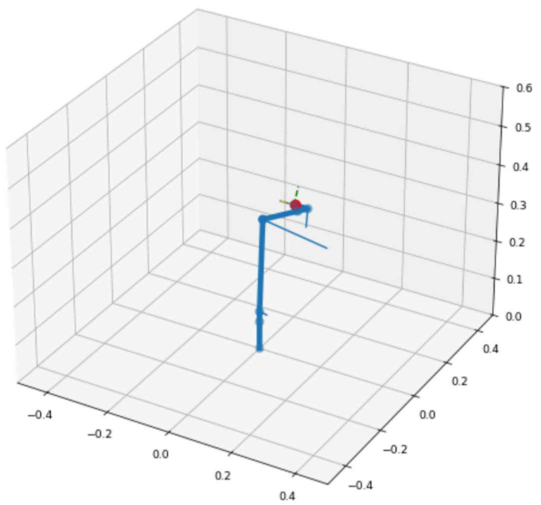
9. References

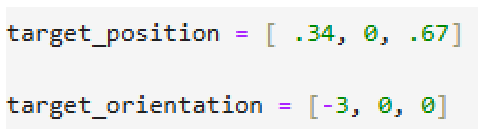
Dejan. “DIY Arduino Robot Arm with Smartphone Control.” *How to Mechatronics*, 11 Sept. 2018, howtomechatronics.com/tutorials/arduino/diy-arduino-robot-arm-with-smartphone-control/#google_vignette.

10. Inverse Kinematics and Simulation (Extra Credit)

To earn the extra credit, we modeled our robot arm in Python using the **IKPY** library (Peter Corke’s Robotics Toolbox for Python). We defined a kinematic chain that matched the physical arm using Denavit–Hartenberg parameters based on Dejan’s CAD design.

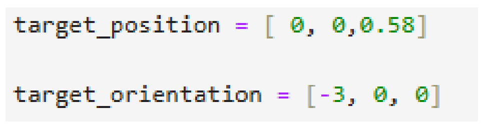






```
target_position = [ .34, 0, .67]

target_orientation = [-3, 0, 0]
```



```
target_position = [ 0, 0,0.58]

target_orientation = [-3, 0, 0]
```

For each of these target points we used IKPY to find the joint angles that move the end effector to that position and orientation. In the first case (left plot), we used target_position = [0.34, 0, 0.67] and target_orientation = [-3, 0, 0], which makes the arm reach out in front and slightly upward. In the second case (right plot), we used target_position = [0, 0, 0.58] with the same orientation, which puts the end effector almost directly above the base. Using this method, we can plug in target positions and orientations to run through and the code will automatically determine where to move each servo. This way we only need target positions, opposed to identifying the correct angle for each servo manually.