ECE 1000 Final Report: Joystick-Controlled Robotic Arm

Jericho Lloyd, Dalton Hall, Gabriel Roessler

Department of Electrical and Computer Engineering
Tennessee Technological University
Cookeville, TN, USA
mjlloyd42@tntech.edu , djhall44@tntech.edu, groessler42@tntech.edu

***Abstract*— For our final project we created a robotic arm to rotate along the x- and y-axis and pick up objects. The project involved the use of a Raspberry Pi Pico W, an analog joystick, three micro servomotors, and a 3D printed arm. This project combined hardware and software components to develop a controlled robotic arm, a simple but foundational project that can be expanded upon to carry out more complex tasks. This report outlines the design, functionality, and background in the making of this joystick-controlled robotic arm.**

Keywords—Raspberry Pi Pico W, Python, Micropython, Micro Servomotors, Analog Joystick, 3-D Printing

I. Introduction

For the final ECE-1000 project, we wanted to create a simple machine that could move around and pick things up using a claw. This project is a foundational step in progressing our skills and learning to develop basic circuits and electronics. The people involved in the final project were Gabriel Roessler—a sophomore Computer Engineering major, Jericho Lloyd—a freshman Computer Engineering major, and Dalton Hall—a freshman Electrical Engineering major.

II. Background

Videos from TopTechBoy on YouTube aided in programming the Raspberry Pi Pico W and connecting the analog joystick to the system[1]. A single line of code, used to initialize the pin that would register the joystick button input, was provided by JC Williams[2], a graduate assistant for our course. For the 3-D print model, we found a model on the site thingiverse.com. The model was made by daGHIZmo, and the model is called the EEZYbotARM[3].

III. Project Description and Formulation

Materials:

1. Raspberry Pi Pico W (RPPW): Central processor that took inputs from the joysticks and controlled the micro servomotors.
2. Servomotors: Main movement component involved with an arm that rotated 180 degrees
3. Analog Joystick: Generic 2-axis joystick used to control the servomotors based on positional coordinates

Diagrams:

The simulated diagram below demonstrates the use of an Arduino instead of a Raspberry Pi Pico W due to the limitations of Tinkercad not having a Raspberry Pi Pico to simulate with. With some adjustments, the Arduino system works identically. A breadboard was used to allow multiple motors and multiple potentiometers (although our actual system used a joystick, it should be noted that the joystick is ran with two potentiometers, and representing the joystick as such is valid).

Figure 1: Tinkercad Simulated System

1
2
3

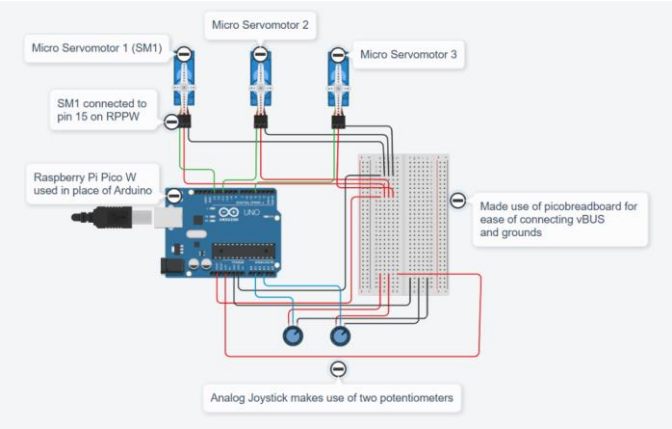
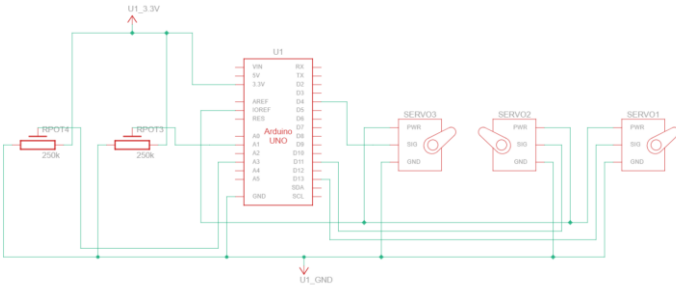


Figure 2: Circuit Schematic of System



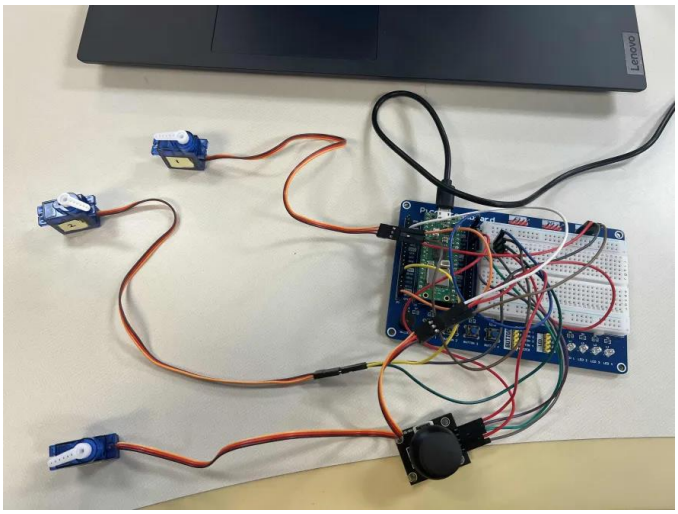
3D Print Process:

The 3-D print process was a simple but tedious process. We found the parts that were required for the model of the robotic arm. After that, we uploaded each file one by one to Tinkercad, and we found that it was best to put everything to 5% scale because 100% would have been excessively big. We thought that it would not take as long if it were set to a smaller scale. The 3-D printer had 18 parts that needed to be printed. This included parts like the base, the arm, and everything that holds the arm together. Due to printer availability and the unavailability of the items to put the print together, the physical arm was unable to be put together and utilized.

Full System:

The picture below depicts our working system. A Raspberry Pi Pico W was used and mounted onto a picobreadboard. We had three micro servomotors, and an analog joystick connected to the breadboard. Power was supplied by the laptop that the Raspberry Pi was connected to. This system in its current state cannot be operated without an external connection to a computer to power and run the code, however, with more time and materials, this could be possible.

Figure 3: Full system, picture taken by Jericho Lloyd



Functionality:

All hardware components involved were connected to the Raspberry Pi Pico W through GPIO (General Purpose Input/Output) pins. These hardware components include the three micro servomotors and the analog joystick.

The analog joystick was connected to ADC pins (pins 26 and 27) that converted the analog signal from the joystick to a digital signal usable by the Raspberry Pi. These ADC pins reported the x- and y- positions of the joystick. From here, these values were converted into x- and y- coordinates understandable to us.

The x- and y- coordinates would be used to determine which and how the servomotors moved.

If the x-value is greater than 10, servomotor 1 moved 180 degrees. If it was less than –10, the motor would move back to 0 degrees. If the value falls between –10 and 10, the motor will stop at its current state until movement criteria are met again.

Servomotor 2 was programmed the same, with the difference being that the y-value was focused on.

The reason for the criteria being less than –10 and above 10, as opposed to less than and above 0, is to allow for easier movement. As there is no change in movement speed, any value above and below 0 initially would cause it to move. When the initial criteria were above and below 0, it was noticed that if someone wanted to move strictly on the y-axis and not x-axis, it was difficult to keep the x-value at a perfect 0. In regular use, x-value would flip between –1 and 1 and servomotor 1 would rapidly move back and forth. Of course, this is not ideal for someone wanting to make a small movement in one direction and accidentally setting off movement in another direction.

With servomotor 3, the register of the button click was the input. When this button was held down and the state was equal to 0, the motor would move the whole 180 degrees. As soon as the button was let go and the state returned to 1, the motor would move back to 0. Currently, this servomotor is not

programmed to save an on and off state and relies on the button being held down.

IV. Discussion and Results

The results of the project were three servo motors that could spin 180 degrees simultaneously using a joystick. One motor responded to the x-axis of the joystick, the other to the y-axis, and the last one toggled by holding down the joystick. Next time it would be great if we had the 3D arm printed so we could see the arm moving up or down and left to right at the same time. The best part of the project was programming the joystick to move the servo motors.

V. Conclusion

This project's purpose was to move a robot arm left to right and up and down. While creating the arm, we learned python, how to 3D print, and how to use a Raspberry Pi Pico. In the end, we didn't achieve the robot arm but created the basic functionality of it using servo motors.

References

[1] McWhorter, P. (2024, March 5). *CALIBRATING JOYSTICK TO SHOW POSITIONAL ANGLE IN MICROPYTHON ON THE RASPBERRY PI PICO W*. Technology tutorials. https://toptechboy.com/calibrating-joystick-to-show-positional-angle-in-micropython-on-the-raspberry-pi-pico-w/#google_vignette

[2] Williams, J.C. (n.d.). *ECE-1000-spring-2024-final-project-insert-project-name/example Micropython Codes/ECE_1000_Joystick_Servo_Example.py at 2D2F07F00D5E50EAA687A12897FC042D79ECC809 · JCWILLIAMS1003/ECE-1000-SPRING-2024-FINAL-PROJECT-INSERT-PROJECT-NAME*. GitHub. https://github.com/JCWilliams1003/ECE-1000-Spring-2024-Final-Project-Insert-Project-Name/blob/2d2f07f00d5e50eaa687a12897fc042d79ecc809/Example%20Micropython%20Codes/ECE_1000_Joystick_Servo_Example.py

[3] Thingiverse.com "EEZYbotARM by daGHIZmo," www.thingiverse.com, September 21, 2015. https:///www.thingiverse.com/thing:1015238