Senior Project

Final Report for

Web-Controlled Robot Arm

In the partial fulfillment of

TECH 4945

Antonio Tinnon & Madison Galloway

Submitted:

5/1/2025

Executive Summary

The primary goal of this project was to create a fully functioning website for the LabVolt Armdroid 1000 robot arm. This robot arm was purchased without a power supply or a teach pendant, and because of this the robot remain inoperable. We were task to develop a website to replace the teach pendant.

The project used a Raspberry Pi 3, an 11-pin Molex connector, and a DC power supply called the Condor HCC15-3-A model. The Raspberry Pi acts as the brains of the project and to host the website through a web server. The website is programmed with HTML, CSS, and JavaScript, and the control backend is programmed with Python using the Flask framework for server-side interaction. The Molex cable also connects the Pi's GPIO pins with the robot arm. The robot arm's motors are controlled through the GPIO pins. The users can choose a motor, type in the number of steps and select the direction of the movement (positive or negative).

This project successfully incorporated web technologies into direct control of hardware systems. All objectives were accomplished, and the UI is very clean and easy to use.

Table of Contents

Introduction

This project combines both of my fields of study, microprocessors and automation/control systems, into one project. Since the robot arm came without a power supply or a teach pendant, we had the opportunity to apply our skills learned to develop a website and use it to control the robot.

There were a few issues. One was that the power supply was not nearly strong enough to move multiple motors simultaneously, so the website was designed around that limitation. Additionally, some of the motors were non-functional and had to be excluded from the interface. The most significant challenge, however, was learning how to develop a website and learn a new programming language to achieve the project goals. I relied on many references to learn Python, HTML, and JavaScript.

Review of Objectives

The objective was to develop a website to control the LabVolt Armdroid 1000. This goal was successful. However, due to the very low power of the power supply, the website was designed to allow only one motor to move at a time. Furthermore, since some motors were no longer working, they were not included in the final interface.

Review of Deliverables

The project deliverables included:

- A working Flask web server hosted on a Raspberry Pi
- A clean, responsive website interface
- Tutorial and help content explaining how to use the interface
- Embedded videos demonstrating the robot's movement
- Fully functional single-motor robot control via user input

Technical Implementation

Hardware:

- **Robot Arm**: LabVolt Armdroid 1000
- **Power Supply**: Condor HCC15-3-A DC
- **Controller**: Raspberry Pi (GPIO used for direct motor control)
- **Cabling**: 11-pin Molex connector
- **Wiring**: Crocodile clips for power connections

Software:

- **HTML/CSS**: For web page layout and styling
- **JavaScript**: For tab handling and user input submission

- **Flask (Python)**: For web server backend and handling POST requests
- **RPi.GPIO (Python)**: For GPIO pin control to move the robot arm

Website Architecture:

1. User interacts with the programming page
2. JavaScript grabs form input and sends it as JSON to the Flask server
3. Flask grabs motor_id, steps, and direction from the request
4. Flask executes a Python subprocess that runs the robot control script
5. The robot control script activates the appropriate GPIO pins to move the selected motor

GPIO Pin Configuration and Functionality

The robot arm's motor selection and movement are controlled via the Raspberry Pi's GPIO pins using BOARD numbering:

BOARD Pin	Purpose
7	Motor select (bit 0)
22	Motor select (bit 1)
18	Motor select (bit 2)
16	Step control pattern (bit 0)
15	Step control pattern (bit 1)
13	Step control pattern (bit 2)
12	Step control pattern (bit 3)
11	Step pulse trigger

- Pins 7, 22, and 18 combine to select one of up to 8 motors using a 3-bit binary value.
- Pins 16–12 control motor stepping sequences using a msteps pattern array.
- Pin 11 sends high-low pulses to execute each motor step.

Evaluation of Plan of Work

The original plan involved:

1. Research web development
2. Research necessary programming languages
3. Designing a full-stack web control system
4. Integrating all parts into one system

5. Testing and iterating based on hardware limitations

All tasks were completed as planned. The only changes were adaptations made due to limited power supply and some motors. The website was modified to match the final list of functioning motors.

Evaluation of Planned Work

The original plan was to research how to develop a website, and successfully operate the robot arm through user input. Each stage of the plan, research, hardware setup, website development, testing, and refinement, were very successfully. While some flexibility was needed due to hardware limitations (such as a limited power supply and motors), these challenges were addressed by changing the UI on the website.

Evaluation Results

The robot arm control system was tested based on usability, reliability, and successful command execution from the web interface. Our testing showed that the system responds well to input and executes movement commands with accuracy and without delay. Each working motor could be controlled individually. There were no significant delays in input recognition or command processing. While the system cannot control multiple motors at once due to the power supply, single-motor operation is stable and dependable. Overall, the results showed the project goals were reached.

Conclusions

This project successfully built a real, working solution to control a robot arm from a website. It demonstrated both hardware interfacing with GPIO and full-stack web development with Flask. Software and design constraints regarding power supply and legacy hardware were overcome. Future work could include upgrading the power supply, giving real-time motor position feedback, and adding sensor integration.

References

- “Raspberry Pi Documentation - Raspberry Pi Hardware.” Edited by Nate Contino, *Raspberrypi.Com*, Jan. 2025, www.raspberrypi.com/documentation/computers/raspberry-pi.html.
- Kohn, Daniel. “LabVolt Armdroid 1000 Project.” *Dan Kohn’s Armdoid 1000 Project Page*, 2009, dankohn.info/projects/armdroid.html.
- Yumpu.com, Lab-Volt Systems Inc. “ARMDROID 1000 Robot Model 5100.” *Yumpu.Com*, www.yumpu.com/en/document/read/32321747/armdroid-1000-robot-model-5100. Accessed 14 Feb. 2025.
- “W3schools.Com.” *W3Schools Online Web Tutorials*, www.w3schools.com/js/js_json.asp. Accessed 24 Mar. 2025.
- “W3schools.Com.” *W3Schools Online Web Tutorials*, www.w3schools.com/html/. Accessed 24 Mar. 2025.

Appendix – Detailed Testing Results

Test Case	Input	Expected Output	Result
1	Motor 1, 50 steps, direction 1	Wrist rotates clockwise	Passed
2	Motor 2, 100 steps, direction -1	Arm moves backward	Passed
3	Motor 3, 75 steps, direction 1	Base rotates clockwise	Passed
4	Invalid input (blank)	Error message	Passed
5	Steps = text instead of number	Input validation alert	Passed

A. Full Form Input Flow (Data Mapping)

UI Label	Field Name	Sent as JSON	Notes
Wrist Input	steps	steps	Sent with positive/negative direction
Arm Input	steps	steps	Input field is shared
Motor Identifier	Hidden	motor_id	Static in HTML form per motor
Direction	Inferred	direction	Based on whether steps is positive or negative

B. JavaScript–Flask Communication Breakdown

Example JSON sent to Flask:

```
{
  "motor_id": "1",
  "steps": 75,
  "direction": 1
}
```

Flask route receives:

```
@app.route("/control", methods=["POST"])
def control_robot():
    data = request.json
    motor_id = data.get("motor_id")
    steps = data.get("steps")
    direction = data.get("direction")
```

Flask passes to subprocess:

```
python3 robot_script_New.py 1 75 1
```

C. Python GPIO Pin Configuration:

```
pins = [7, 22, 18, 16, 15, 13, 12, 11]
GPIO.setmode(GPIO.BOARD)
```

D. Screenshot Previews:

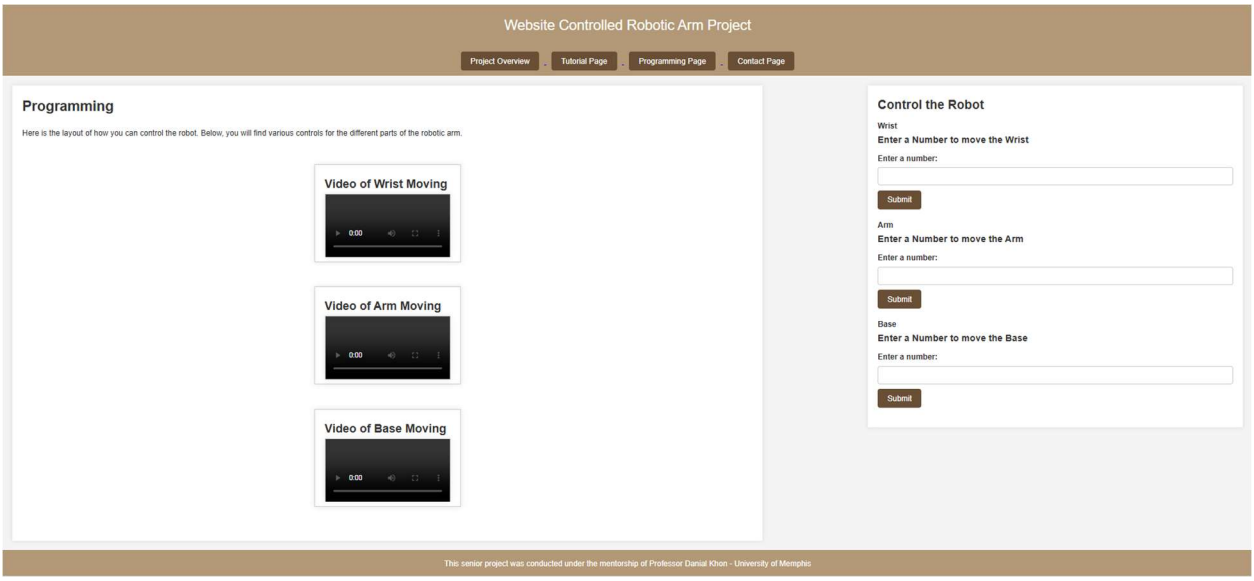

Figure D1: Main Programming Page Interface

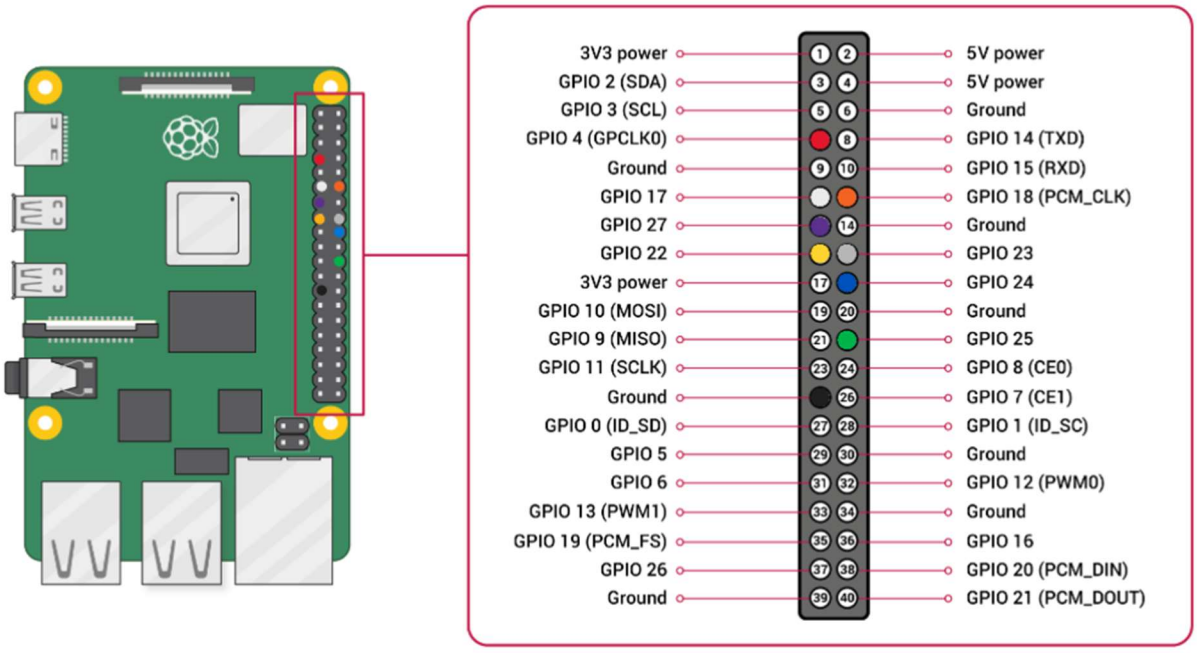

Figure D2: Raspberry Pi's GPIO Pin layout

E. Additional Notes on Testing

- Robot was physically connected and movement visually confirmed
- Step directions and values were tested
- Confirmation and error messages were tested using incorrect inputs
- All testing occurred with live hardware and current Flask updates
- Project folder includes a backup of the last fully working updates