MEAM 5100 Lab 4.2- Team 7

Kevin Paulose (63031616), Tejendra Patel (35371691), Saurav Agrawal (58987928)

November 21, 2023 (4 out of 5 late days used)

1 Description of mobile base approach (with photographs)

Our mobile base is crafted with a sophisticated 4-wheeled mechanism. The decision to adopt a holonomic mobile base with mecanum wheels signifies a strategic commitment to unparalleled agility. Mecanum wheels, distinguished by their oblique rollers, provide seamless movement in all directions with exceptional precision. This design facilitates instant changes in velocity which prioritizes maneuverability. The mecanum wheel configuration not only enables lateral gliding, axis rotation, and complex trajectories but also ensures superior mobility, making it an optimal choice for dynamic applications. Contrary to the usual 60mm diameter mecanum wheels, we have used 80mm diameter mecanum wheels to bulk the design and make it sturdy on the track. The design of the bot is shown below-

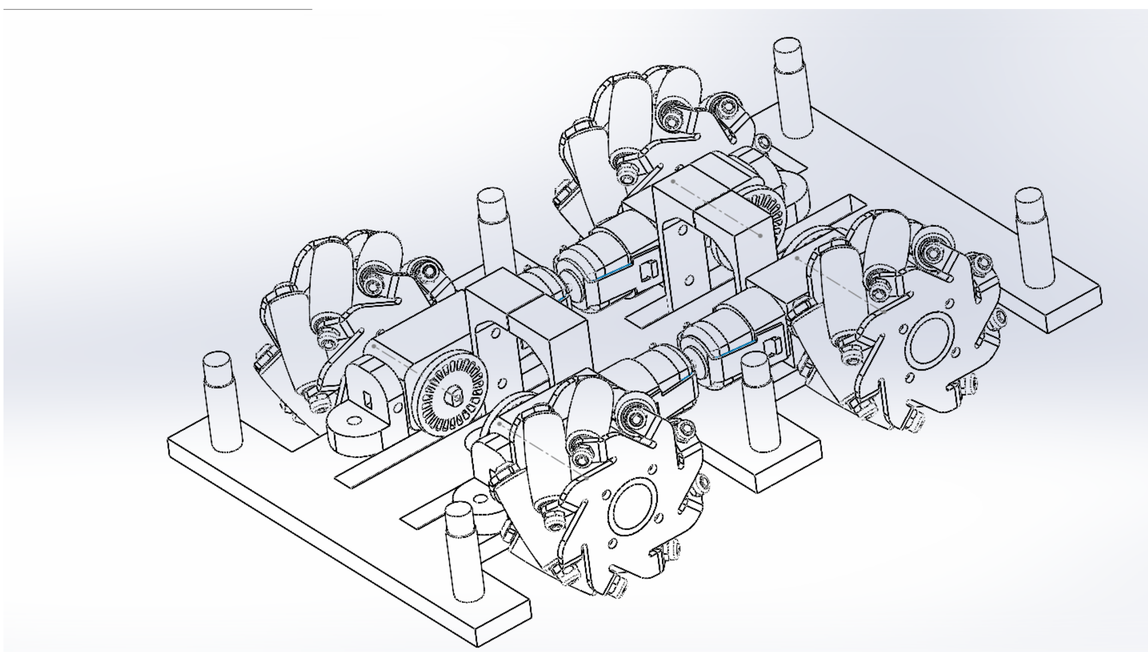


Figure 1: SolidWorks Model of Base chassis with mecanum wheels, motors and encoders mounted along with other supporting attachments (for stability)

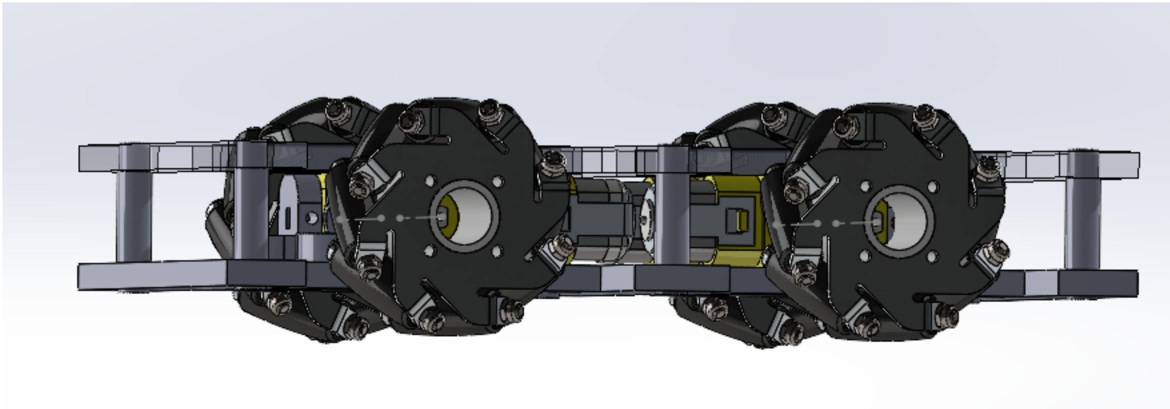


Figure 2: Upper chassis attached (full vehicle model)

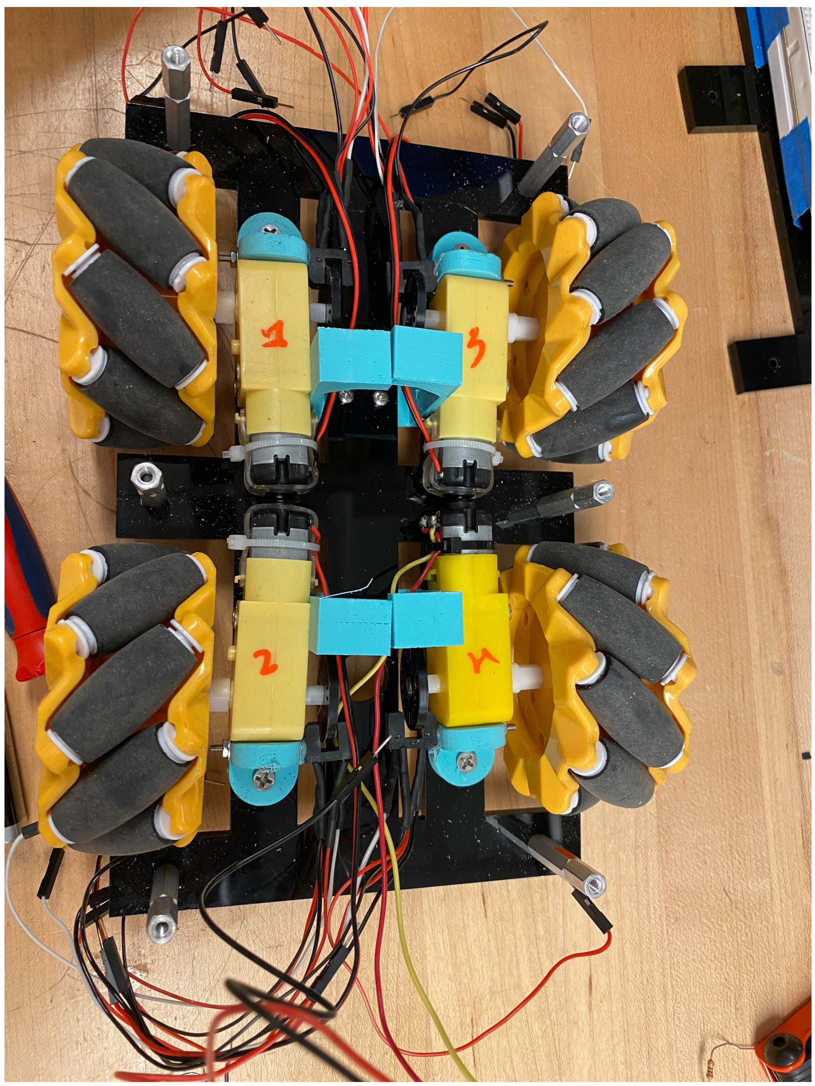


Figure 3: Base chassis of the robot

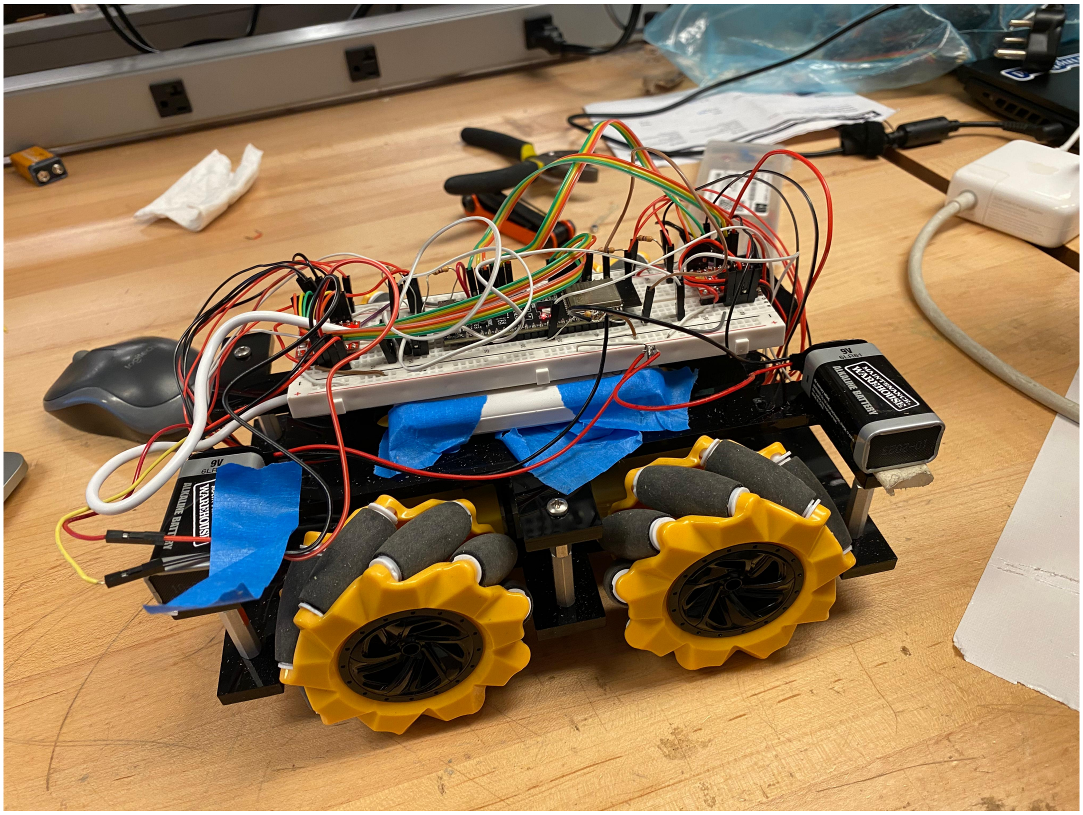


Figure 4: Whole robot

Dimensions:

- Mecanum wheels: 80mm diameter
- Vehicle Dimensions: 230 mm x 140 mm x 100 mm
- Motor: TT DC Gearbox Motor (4 units)
- Spacer and back support: 4 each (3d- printed)
- Base and Top frame: laser cut $\frac{1}{4}$inch Acrylic (black color)

2 Discussion of performance (list your race time) and list of improvements

The recorded race time of **57.91 seconds** reflects the current state of our robot. While this achievement is noteworthy, there is always room for improvement to optimize speed, precision, and overall performance based on our observations-

- **Mechanical Design:** One area of improvement lies in the mechanical design, specifically the tolerance of the wheel alignment. Enhancing the precision and stability of the wheel alignment system can contribute to a straighter trajectory, minimizing unnecessary deviations during the race. This adjustment aims to improve the overall efficiency of the bot and reduce time losses resulting from imperfect alignment.
- **Electronics and System Robustness:** To bolster the robustness of our robot, integrating Printed Circuit Boards (PCBs) and Battery Management Systems (BMS) could be pivotal. PCBs can streamline the electrical connections, minimizing the risk of loose wires or faulty connections that may impact performance. Additionally, a BMS can optimize power distribution and ensure the longevity of the battery, thus maintaining consistent performance throughout the race.

- **Safety Measures:** Implementing a kill switch mechanism is a crucial safety enhancement. This feature can swiftly terminate the program in the event of a power surge or any unforeseen issues, preventing potential damage to the bot and ensuring the safety of both the bot and its surroundings. This proactive measure aligns with best practices in engineering design and addresses potential risks associated with unexpected power fluctuations.

3 Your design must include a form of relative position feedback (either encoder on wheels or mouse sensor) and feedback control (P or PD or PI or PID). Submit two videos of the robot being commanded to go straight for 1m. In one turn control off and show that a push midway disrupts the motion. In the 2nd video show the control self-corrects any midway push.

We have implemented a Proportional feedback control using encoders attached to the yellow TT DC motors and optical sensors (QVE00120) to get motor RPM readings. We discussed and found that there was no need for *Kd* gain factor as also told by professor, also on seeing the graph of proportional control we saw that it was working pretty accurately and there was no drift with time so there was no need for *Ki* integral part.

Our problem statement for the control law was velocity control. Since the PWM values for each motors varied according to it's condition, battery potential and other minor factors, a velocity control law which enabled desired values for each motor for a desired motion. So we got a desired_RPM and get feedback from encoder in terms of estimated_RPM and calibrating *Kp* to get the calculated PWM for each motor. A push midway disrupts the velocities of each motor. The control law helps in the motors to re-calibrate their desired speeds and go on with the same motion as before. Trajectory control could not be done as there was no localization feedback involved anywhere in this robot project.

When we initially implemented P control for the encoder feedback, we encountered a significant amount of noise in the system. Recognizing the need to mitigate this noise and attain a smoother feedback signal, we experimented with various filtering algorithms. The first attempt involved the application of a moving average filter, a relatively straightforward technique that computes the average of a set of recent data points to smooth out fluctuations in the signal.

While the moving average provided some improvement, we aspired to achieve an even higher level of performance. We decided to leverage the capabilities of the open-source SimpleKalmanFilter library. This library implements the Kalman filter, an recursive algorithm designed to estimate the state of a dynamic system amidst noisy measurements. The Kalman filter is particularly advantageous in scenarios where uncertainty and noise in sensor readings pose challenges.

By integrating the SimpleKalmanFilter library into our control system, we experienced a notable enhancement in the smoothness of the feedback signal. The Kalman filter's adaptive nature, which combines information from current measurements and predicted states, proved to be highly effective in producing a more accurate and less noisy output.

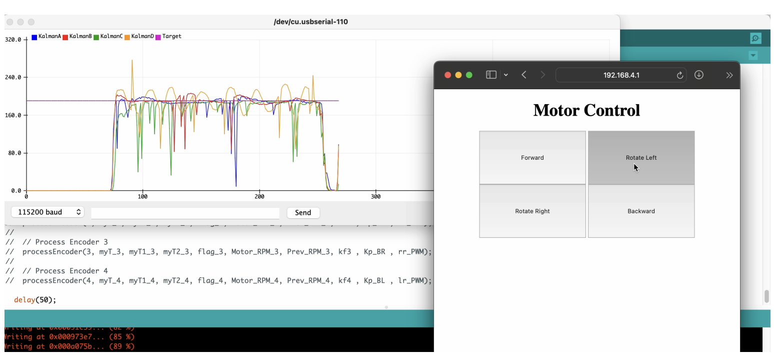


Figure 5: P Control of all 4 motors achieveing target RPM.

Videos demonstrating our successful P feedback control law is as follows-

Control is OFF, push midway disrupts the motion- YouTube link

Feedback control output chart- YouTube link

Control is ON, push midway is self-corrected- YouTube link

4 Final version of code used on mobile base and input ESP32. (include code for race and 4.2.2.1 if different)

The code used in the race was without the Proportional(P) feedback control. We did this as the noise generated from track friction and other moving parts of the robot made the feedback control inaccurate over time. Adjusting the PWM logic helped us clock the best time for the race. However, for 1m distance, we were able to successfully demonstrate the feedback control working. We have submitted both code, named Team_7_Lab4.2_race_code (without feedback control) and Team_7_Lab4.2_PID_code (feedback control code).

3 laps race circuit (similar to actual race)- YouTube link

5 Bill of materials

Item name	Quantity	Item Procured from	Total price
TT yellow stock motors	3	GM lab	-
TT yellow motors	1	Friend	-
Mecanum wheels	4	Amazon (PEFS)	$ 24.30
Motor Driver	3	Amazon (PEFS)	$ 9.99
Voltage regulator	2	Amazon (PEFS)	$ 5.49
5000mAh Mady Power Bank	1	GM lab	$ 7.00
ESP32 S2	1	GM lab	$ 8.00
QVE00120 photo Interrupter	4	GM lab	-
Encoder	4	GM lab	-
9V alkaline battery	8	Friend	-

- Total group funds: $ 150.00
- Total cost of purchased parts: $ 54.78
- Balance funds amount : $ 95.22

6 Circuit diagrams

The complete circuit diagram of our robot is as follows-

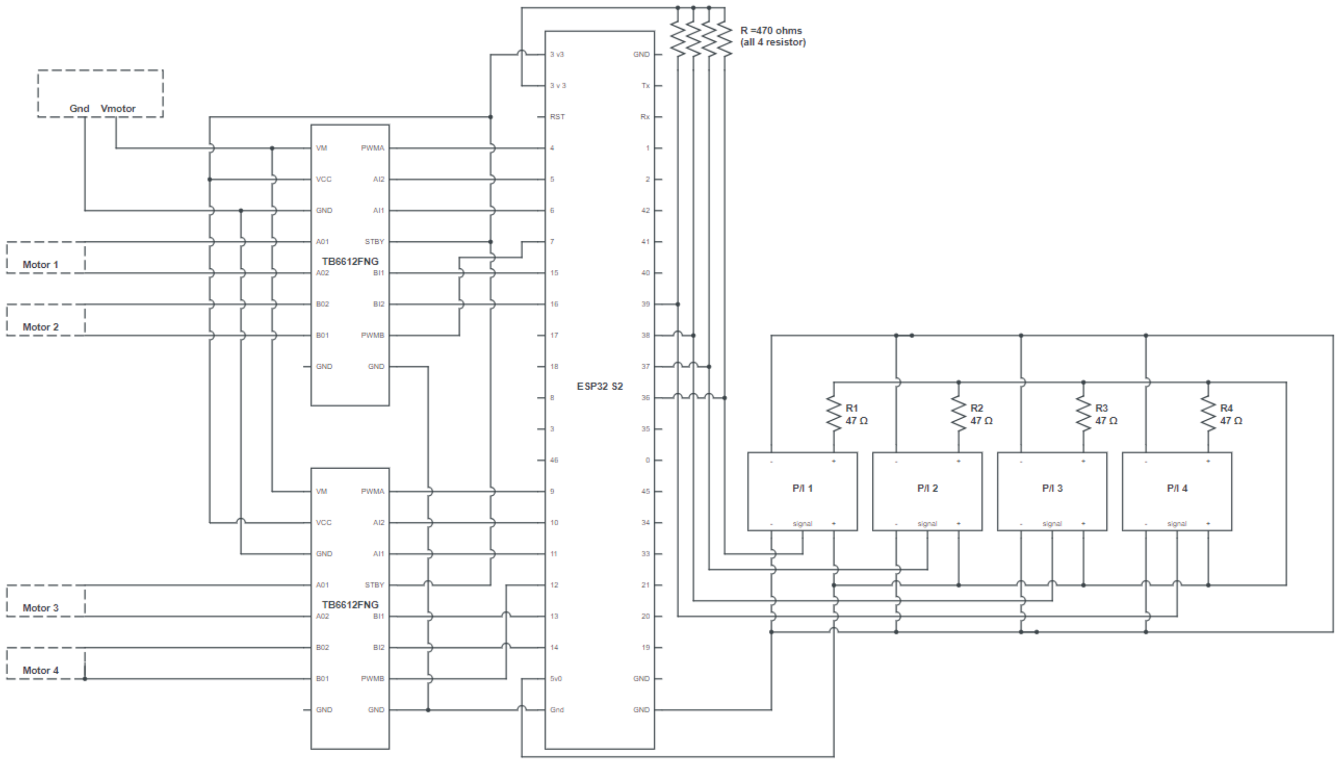


Figure 6: Robot circuit diagram

7 If work was divided between team members, indicate who was responsible for which part.

In our collaborative project, each team member played an integral role, contributing equally to ensure the successful completion of the task. The primary task division we Design and Fabrication- Saurav Agrawal, Code- Tejendra Patel and Circuits, Soldering, Debugging- Kevin Paulose. From brainstorming ideas to executing the plan and finalizing the details, every team member demonstrated a strong commitment to their assigned tasks. Each member's dedication and willingness to contribute ensured that the workload was evenly distributed, ultimately leading to a cohesive and well-executed project.

8 (Extra Credit) Solder all circuits and use connectors to connect all off-board parts (e.g. motors) with twisted wire cabling. Use sockets for the ESP32C3 and other active components.

Since our design involved 4 yellow TT DC stock motors, QVE00120 photointerruptors, we had a lot of wires, specifically 20 from the photointerruptors and 8 from the motors. And then we also had 2 motor drivers, each having another big set of wires. Hence we used the ESP32 S2 to compensate for the number of board pins. To be on the safer side, we did our race using connections on a breadboard. Once we were timed, we moved on to the perfboards in place of breadboard. Being new to perfboard soldering, we did some iterations to come up with the best perfboards for our circuit. Below are the final soldered perfboard-

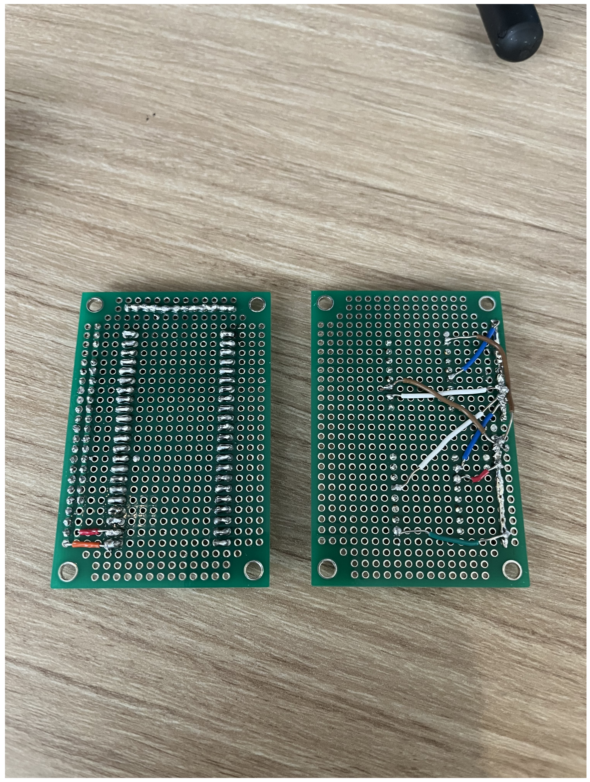
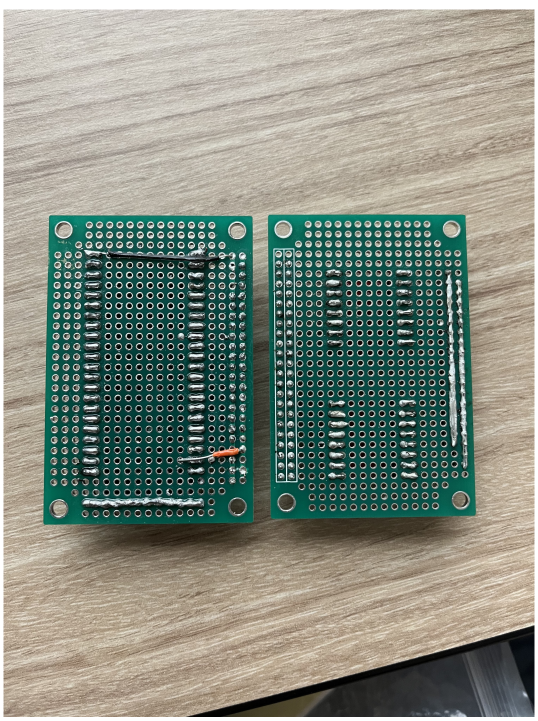
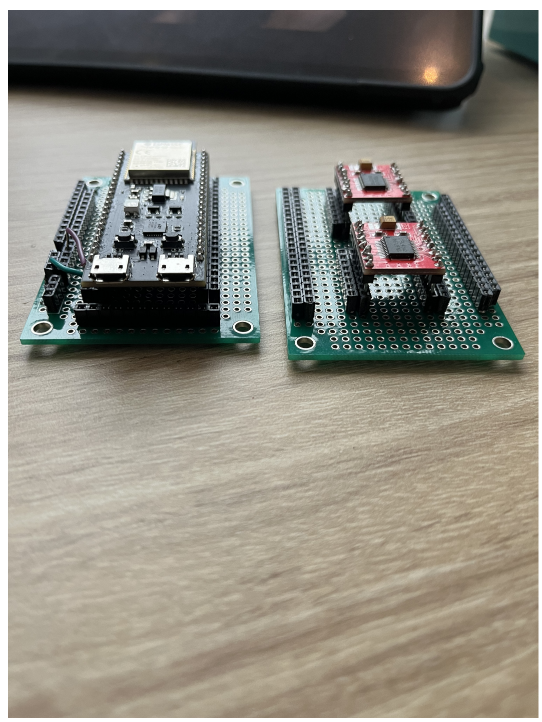


Figure 7: Initial perfboard iteration (left for ESP32 S2 and right for 2 motor drivers)



(a) Final perfboard (left for ESP32 S2 and right for 2 motor drivers)



(b) Perfboards with ESP32 and motor drivers mounted

Figure 8: Final perfboards used for the robot

We know how to improve our perfboard by using twisted wire cabling, crimping, etc. and we will be doing that for the final project.

Driving the robot using soldered perfboard connections- YouTube link

References

- https://dronebotworkshop.com/mecanum/
- https://www.arduino.cc/reference/en/libraries/simplekalmanfilter/
- https://electricdiylab.com/how-to-connect-optical-encoder-with-esp32/