Final Project - MEAM 510

Anokhee Chokshi, Raima Sen, Sofia Panagopoulou

1 Functionality

1.1 Approach to winning the game

Even though initially we were planning on working on the competition functions, we ended up having little time so we could not reach our expectations. Some of the ideas we had and implemented (or thought about implementing) are listed below:

- The primary reason for choosing mecanum wheels was that the robot could change directions and move easily during the game. With a mecanum wheeled drive, the robot could even rotate clockwise or counterclockwise at its position and so we could avoid mounting servo motors on the body for scanning (e.g. for frequency detection).
- The structure of the robot was created such that the photo diodes for the vive are mounted on a high level, in order to get better results.
- Adjustments were made in the design such that the IR sensor could be at the same level as the beacon LEDs for increasing the efficiency.
- A protective design of the robot was made so that it wouldn't be damaged if it hit the walls or another robot.
- We had intended to create a webpage for switching between modes during the game. We had integrated 3 modes for the checkoff, but our plan was to make a screen for the game.
- Notches were made in initial design for grabbing the cans. However, this model was not used during the game.
- We had thought of using an electromagnet for picking up the cans, since there were talks of them having something magnetic at one point. We had even thought of using a suction mechanism. However, we did not implement any of them.

1.2 Approach to the functionalities and Analysis of what worked and what didn't

1.2.1 Wall following

- For the wall following function of the robot, we decided to use two ultrasonic sensors at the side of the robot and one TOF sensor at the front. Since the TOF sensor is more accurate than the ultrasonic (detecting range of 25 degrees versus 75), it is used to detect an upcoming corner. The two ultrasonics are used to help the robot keep a specific distance from the wall. We selected to have two of them instead of one because of the robot design. Due to the mecanum wheels and the size of the robot, it is important to make sure the robot's rotation remains stable. So the plan was to compare the two ultrasonic's readings.
- Along the way, we realised that one ultrasonic did not have a great performance so we only used it for backup. The rest of the code relies on the other ultrasonic and the TOF.
- The results were above expectations and we believe this functionality was the most accurately represented by our robot.

1.2.2 Frequency detection

- For the beacon detection, the hardware was selected based on Professor Mark's schematic and our testing. We made an identical circuit and then concluded on the most reliable values of the resistances and the voltage inputs.
- While testing the circuit with an oscilloscope and a frequency generator, the performance was very satisfying - especially compared to our lab 2 circuits. Transferring the circuit to the robot, the performance remained excellent.
- We started having problems when introducing the robot movement to this functionality. The mecanum wheels cause vibrations to the robot while on the field and the mounting of the IR sensor was not ideal due to our limited time. The vibrations which were more apparent at the IR sensor's height, caused it to move a lot and therefore, not be aligned to the beacon LEDs.
- The combination of these two drawbacks led to poor overall performance of the frequency detection function. The robot had to be approximately one meter away to be able to detect and follow the desired frequency.

1.2.3 HTC Vive Tracker Detection

- Keeping the voltage divider resistance ratios same, the resistance were increased to a higher value.
- The cathode and anode of the photodiode were interchanged and the performance of the circuit was inspected in both the cases.

- This inspection was made easy due to the robust nature of the circuit wherein green connectors allowed this interchangeability.
- It was initially assumed that 2 vive detection circuits were required to provide accurate navigation, as we could then examine the orientation of the robot too.
- However, with the use of mecanum wheels, this requirement was no longer needed.
- At the GM lab, due to the presence of external lights, the photo-diode did not provide accurate readings. We believe this was probably due to the use of high gains in the voltage divider, however this is unlikely to happen. This implies that the photodiode was extremely sensitive.
- The vive circuit gave readings close to range of 8000 right under the Vive tracker, where the coordinates ideally should've been 4000.
- However, when the testing was shifted to Wu and Chen auditorium, due to the dim lights, the performance of the vive tracking circuit became significantly more accurate. Now the range of values were 4000 as expected.
- Since initially, the coordinates emitted by the can had their X and Y coordinates switched at the Wu and Chen Auditorium, we realised calibrating the robot X and Y coordinates to match the frame of the cans would be a sane way to proceed autonomous navigation for the game.
- Using the 2D homogeneous transformation matrix, we mapped the X and Y coordinates of the robot in the can frame and then proceeded to calibrate the wheel motions.
- This was unnecessary, as the can coordinates were fixed for the competition.
- However, the robot was able to locate coordinates in 8 directions accurately in the low light setting and using mecanum wheels.

1.2.4 Manual control

- A major part of the manual control interface was adapted from Lab 4.
- The following elements were present in the UI :
  - 8 buttons to control the direction of motion
  - a STOP button
  - Duty cycle slider bar to control the speed of the robot
  - 2 slider bars to input the X and Y coordinates for the robot to go to using the Vive tracker
  - 3 buttons to switch between manual control, beacon tracking and wall following modes

- We initially intended to use a joystick to have a more seamless interface. However with the use of custom MEAM510.h and MEAM510.cpp, we were unable to get the xhttp part right so the values were not getting sent from the webpage to the subroutines in the arduino functions.
- Ideally, the required X and Y coordinates must be taken as input from the user, to go to a certain location. The use of slider was simply due to the lack of time.
- For switching between various modes, the interface should have had all features, not just 3 (manual control, beacon tracking and wall following modes). We could not include the vive detection button and gripper arm button, again due to mismanagement of time on our part.
- In Lab 4, we had used the AP mode with our own IP address for TCP/IP communication. However, in the final project, due to the use of UDP for transmitting and receiving X and y coordinates from the vive tracker, we had to use station (STA) mode.
- For this, we had to use the common router to get connected to our web interface. However, for some reason, we lost connection to the interface several times unknowingly. This issue was absent when we were working in the AP mode for Lab 4.

2 Mechanical Design

2.1 Intended vs Actual performance

- We were using the model from Lab 4 since we had some issues with our new design. We thought that the robot was quite sturdy. For the game, we had placed the top cover at a height such that it could push cans to the box, since we did not have scope to integrate a gripper.
- Since the robot was drifting, we calibrated the motors to control it.

2.2 Things that we tried but failed

- A new model was designed and fabricated for the final project as we realized the shortcomings of the model built for Lab 4. This design was made so that the cans could be fit easily into the notches during the game. However, since this was a new configuration for the arrangement of the mecanum wheels and since we could not find much literature about this out there, we had to switch back to Lab 4's model as we couldn't calibrate the motion and running out of time.

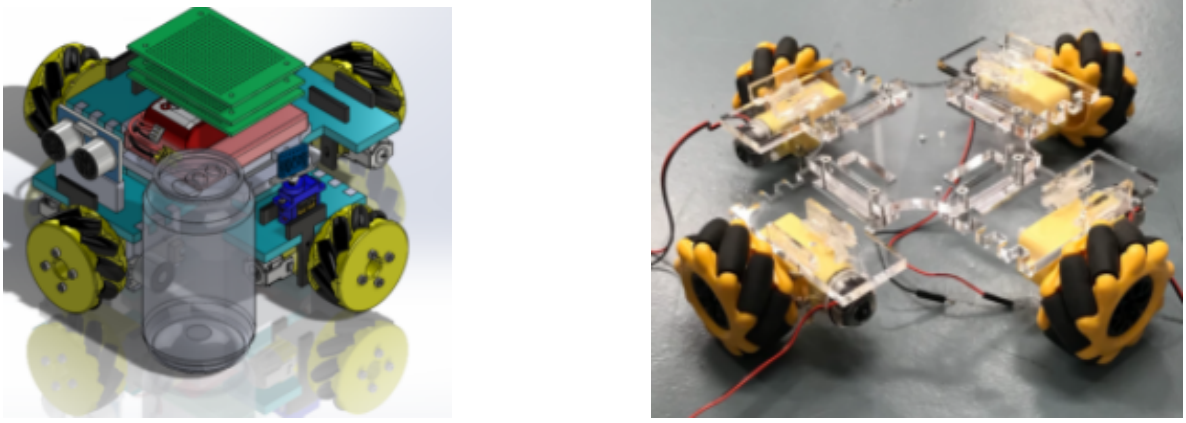


Figure 1: Design #2 model

3 Electrical Design

3.1 Performance

- We used 3 perfboards for all the electrical connections. The first one contained the circuit for the IR sensor. The second one had the vive circuits and the dipswitch. And the top most one had the ESP32, H-bridges and motor connections.
- In order to reduce the number of pins required, since we were only using one ESP32 board, we used inverters for the direction pins of the motors. After including the inverter, they did not work initially, since the pullup resistors were missing.
- We used green screw mount terminal blocks, so that we could change the pins and connections without desoldering. Since we were using almost every pin of the ESP32, we had a lot of wires coming out and hence it was a good idea to use these as connectors.
- At certain points, using old wires resulted in noisy data. However, on changing the wires, we got much cleaner signals.
- We had soldered connector pins to the perfboard, so that we could remove the ESP32 whenever it is needed. However, in the process of removing and placing it a number of times, the pins started bending and we even broke the antenna of one of the ESPs in the same process.

3.2 Things that we tried but failed

- Our initial soldering became very complicated as we kept adding new circuits to the robot. We chose to resolder the majority of it in order to have easy access and better color coding for debugging. We also used many green connectors between perfboards so that we would be able to move each perfboard independently.
- We tried upgrading our voltage from 5V (power bank) to 7.4V (lipo). Along the way, we burnt four microcontrollers and even though we found some reasons that could have caused each of them, the common factor was the LiPo battery. This is why we chose to switch back to the portable charger since the speed of the robot was not critical for the project.
- We fried one of the ESP32s because of an error made while writing the code. A web interface was being used to control the direction and speed of the motors. One button was linked to 3 functions, which should not have been the case. So pressing that button caused a load on the ESP because it tried to do 3 things at once.
- We broke the antenna on an ESP while removing it from the perfboard.

4 Processor architecture and code architecture

4.1 Block diagram

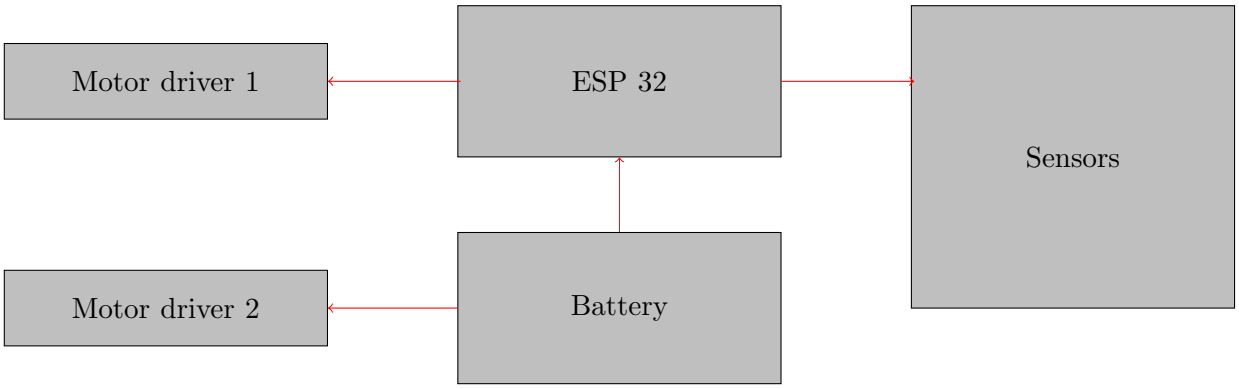


Figure 2: Block diagram for powering

4.2 Software approach

- The general approach for coding was to start from as simple code including the external elements (sensors) and through testing, slowly build it to full functionality.
- Incorporating the functionalities described below, we added three mode buttons to the web interface. Each button was setting a variable mode to a specific number and this helped separate the functions. We incorporated the Wall Following with the Frequency Detection and the Manual Control. The "Stop" button from the manual control was bypassing all the mode numbers, as a safety measure.

4.2.1 Wall Following

- For the incorporation of the TOF and the ultrasonic, we used the two Arduino built-in libraries VL53L0X and NewPing respectively. Since we relied on the one of the two ultrasonics, the code was making the robot move forward and checking if the distance of this ultrasonic became more than 20cm or less than 5cm. It was placed at the front right corner so in the first case, the robot was rotating clockwise, and in the second, counter clockwise. The second ultrasonic which was placed at the bottom right corner of the robot, was included with an OR statement in the second case (if front ultrasonic is less than 5cm OR second is more than 20cm). When the TOF detected something within 20cm, then the robot was instructed to rotate for 1 second.
- The sensor measurements and the functionality code were taking place in the loop function.

4.2.2 Frequency Detection

- The code included the usage of an interrupt. Whenever a rising edge was detected by the IR sensor, the interrupt is triggered. We subtract the previous time of trigger from the current time and then set the previous equal to the current. That way, we have the duration between the two rising edges in counts. In the loop function, we check if this duration is less than 1500 (700Hz) or more than 42000 (23Hz) and make the robot move forward. In any other case, the robot is rotating clockwise, scanning its surroundings. These values were obtained starting from the equation: counts = system frequency/(prescaler*desired frequency) and conducting a lot of testing.

4.2.3 Vive

- Since we were working at the Wu Chen auditorium and the last cans were made differently, we were getting a different set of readings for the cans. So we implemented a transformation of co-ordinates in our code, so that the robot frame matches the can frame.

4.2.4 Manual Control

- For the manual control code, we used the button and slider code on Canvas and combined them. We then added several more buttons for the movement of the robot.
- Taking advantage of the mecanum wheels, the possible movements were: (1 is front left, 2 is front right, 3 is back left, and 4 is back right)
  - forward left (2,3 forward)
  - forward (1,2,3,4 forward)
  - forward right (1,4 forward)
  - left (2,3 forward, 1,4 backward)
  - stop (1,2,3,4 stop)
  - right (1,4 forward, 2,3 backward)
  - backward left (1,4 backward)
  - backward (1,2,3,4 backward)
  - forward right (2,3 backward)
- All the configurations were done in the setup function, and then each of the handlehit functions linked to each button was writing the motor speeds and directions. The slider was linked to the duty cycle of the ledc, and therefore, to the speed of the motors. Since the slider only changed the global variable of the speed, no change was visible until a button was pressed.

4.3 Things that we tried but failed

- For the vive web interface, we had a slider to input the x and y positions. However, an input text box would have been a better choice.
- As mentioned earlier, we were trying to work on a joystick for manual control instead of the buttons. It would have resulted in the robot to move in any direction and changing the direction could have been smoother.

5 Retrospective

5.1 What you feel was most important that you learned

- The most important thing that we learnt was to be careful of the power management of the circuit and to conduct all the calculations beforehand.
- Also, in a project that is expected to significantly increase in size, it is very important to keep all the equipment and documentation organized and up to date.

5.2 What was the best parts of the class

- The best parts of class were the lab assignments and the cooperation between classmates.
- Also, the freedom we gained slowly during the semester gave us more and more responsibilities and excitement.
- The final competition would definitely be one of the best parts of class if our preparation was better.

5.3 What you had the most trouble with

- We had the most trouble with trying to incorporate all the new things we wanted from Lab 4 to the final project. It was difficult to keep up with "what broke what", and therefore, we redid many parts of the project that might have been unnecessary.

5.4 What you wish was improved

5.5 Anything else about class

6 Appendix

6.1 Bill of Materials

Item	Quantity	Price
USB cables	2	$9.64
Ultrasonic sensor	1	$3.95
TOF sensor	1	$12.90

6.2 Schematics

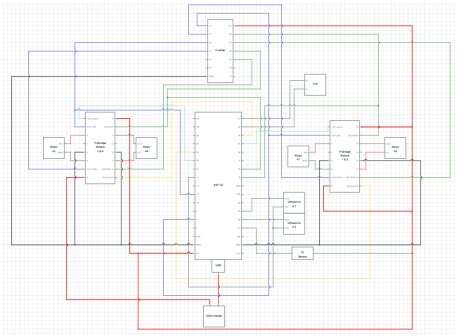


Figure 3: Circuit diagram for motor drivers, TOF and ultrasonic sensor

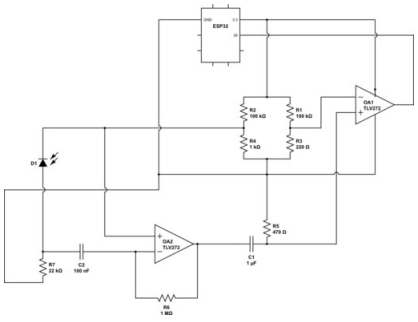
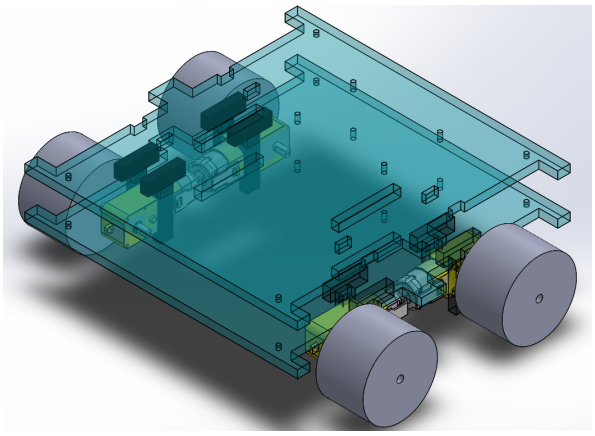
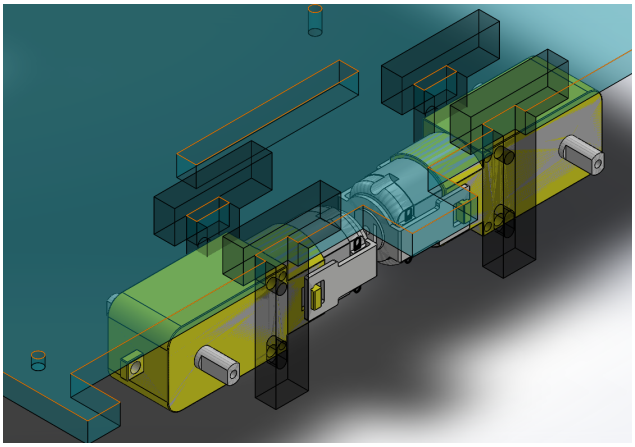


Figure 4: CIrcuit diagram for vive

6.3 Photos



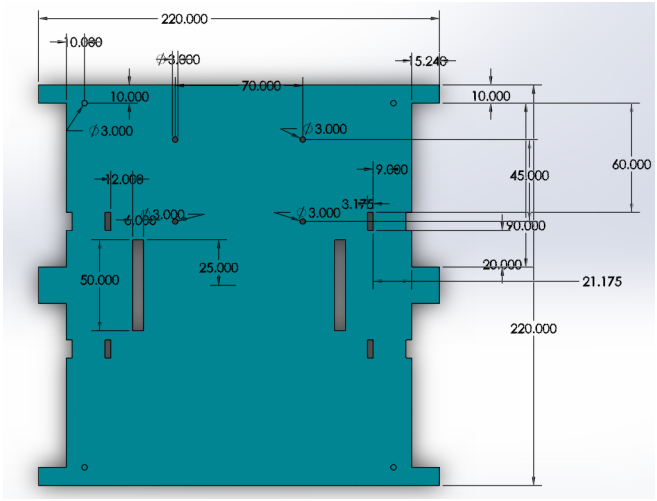
(a) Isometric view of the robot



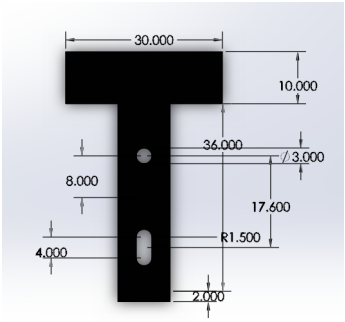
(b) Motor attachment using T's

Figure 5: Assembly of the robot

6.4 CAD drawings



(a) Dimensions of the base and cover



(b) Dimensions of the T used

Figure 6: Assembly of the robot

6.5 Datasheets

Components used
TOF sensor
Ultrasonic sensor

6.6 Videos

- Wall Following
- Beacon sensing