Final Report - Concept, Design, and Execution

Moyosola Omole

Craig Reimer

Ryan Tierney

Executive Summary

This project called for the creation of an autonomous driving robot that would successfully complete a course around the Stevens campus with obstacles and constraints given. The main components of building the robot dealt with the hardware and software as we had to design an original robot body, as well as compile original lines of code that directed the robot to run the course and navigate through obstacles. We were given limited materials, with strict constraints dealing with time limits and hitting targets. In terms of the hardware of the robot, the 3D printed design had to account for the WeMos Board, battery, protoboard, GPS float, ultrasonic sensors, and an OLED display. An additional requirement was that the 3D printed design had to be printed in under 4 hours. The original design was simple, not accounting for all of the required objects. The design was a square shape, mirroring the dimensions of the chassis board and mounting holes. We accounted for three ultrasonic sensors, having pockets on three sides of the square shape for them to fall on. (See Figure 1:1) As we gradually finalized our design, we had to account for the rest of the materials, so we prioritized utilizing surface area in an efficient fashion to fit all of the objects. The finalized design essentially had two parts that fit together when assembled on the robot. Still taking direction from the mounting holes, we created a design that accounts for the sensors and protoboard in one part, and the WeMos Board and GPS Float in the other, with both the protoboard and WeMos Board standing vertically.(See Figure 1:2 and Figure 1:3) The final design proved to be sufficient as we were able to fit all of the required objects and its compact and effective design allowed us to meet the time limit requirement. However, the design failed to account for wiring issues which proved to be a problem in the final evaluation.

Introduction and Project Objectives

The basis of this project was comprehending the concept of engineering design. We were introduced to engineering design being defined as “a thoughtful process for generating plans or schemes for devices, systems, or processes that attain given objectives while adhering to specified constraints”. We were forced to implement this idea and understand our own given design objectives as well as design constraints. The design objective was the process of successfully building a robot that will hit all the targets on the given course in a certain amount of time. Submitting to the given design constraints was heavily emphasized as failing to yield to the limits would result in a failure. We had multiple design constraints such as the print time limit, accounting for the required objects, and the time limit for completing the course.

Design Requirements

1. Must use the LiDAR sensor and at least three, but up to six ultrasonic sensors for the path planning of the robot.
2. Robot hardware must be able to hold WeMos Board, battery, protoboard, GPS float, ultrasonic sensors, and an OLED display.
3. The breadboard cannot be used. Must use Printed Circuit Board to supply power.
4. Robot is expected to meet specific parameters: a pole diameter of 0.31 inches, a robot height of no more than 5 inches, and no tape or glue to secure the parts
5. Must follow a color coding for the power line

6. Must include a 3 seconds time delay at the end of the setup() function in the Arduino Sketch program.
7. The robot must stop for two seconds when it reaches a distance within 10 cm of the target.
8. Must add obstacle avoidance and target reaching algorithm to given code
9. To get full score, the robot must complete the task within 3 minutes without crashing, reaching all 4 targets in a given order
10. We were given a time frame of 14 weeks to work on and finalize a design

Concept Design

The original design that was created, (See Figure 1:1) only accounted for the Road Test, so it focused on holding the WeMos Board and battery. The Road Test called for us to design the robot to complete a basic driving test. The robot had to successfully go straight, stop, turn, and detect obstacles such as walls. We had to account for one or more

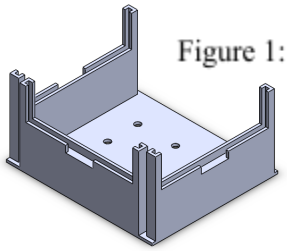
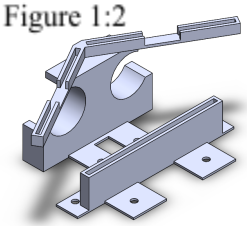


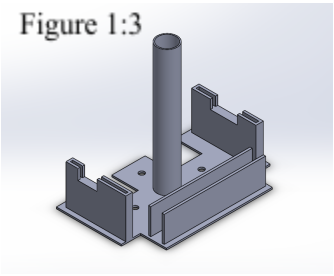
Figure 1:1

Figure 1:2



ultrasonic sensors for obstacle/wall detection purposes. We decided to account for three sensors in order to prepare for the finalized design. This design had to have a maximum print time of 2 hours and 30 minutes, so we focused on simplicity. This design ultimately completed the Road Test successfully while meeting the print time limit. When thinking about the finalized design, we considered breaking up the design into two parts (See Figure 1:2 and Figure 1:3) having it almost fit like a puzzle due to difficulties in the complexity of an updated 3D printed design. We essentially stuck with this concept and committed to executing it successfully as we didn't have any other options to consider. With us having more parts to account for in this design, the idea of separated hardware proved to be more advantageous as we were able to prioritize each part holding the required materials, not only to ensure we included everything but also to ensure they were accounted for in an efficient manner. The finalized design also revealed its superiority in terms of print time as the original attempt to print a single design had too high of a print time, going over the 4-hour limit. This design also fixed the issue with the placement of the sensors as we had problems with the sensors interfering with the wheels of the robot. The elevated and angled placement of the sensors made the two parts of the robot coincide together efficiently. We also took into consideration weight and the center of gravity as we made sure the final design didn't weigh down the robot, possibly slowing it down and hindering its performance.

Figure 1:3



Final Design

Figure 1:4

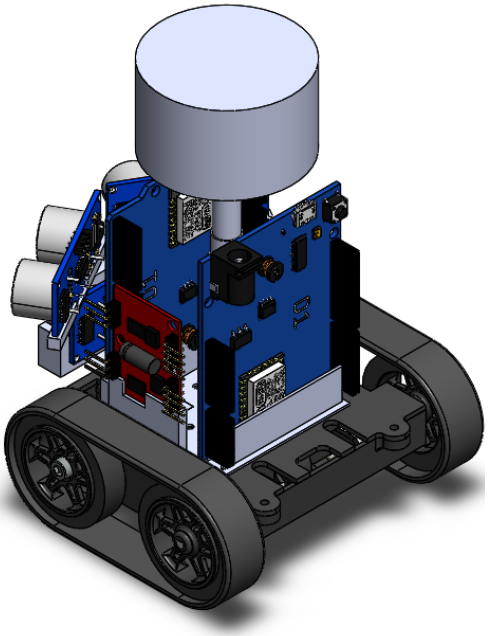
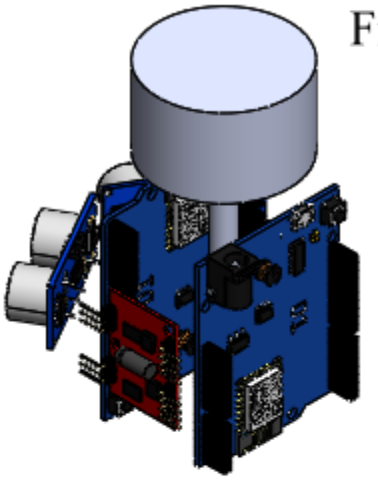
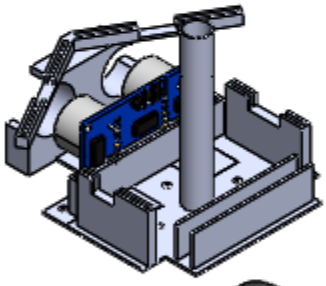
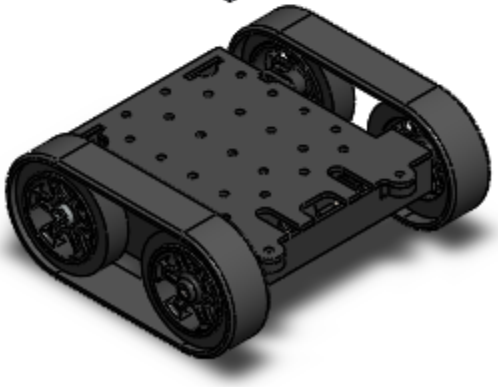


Figure 1:5







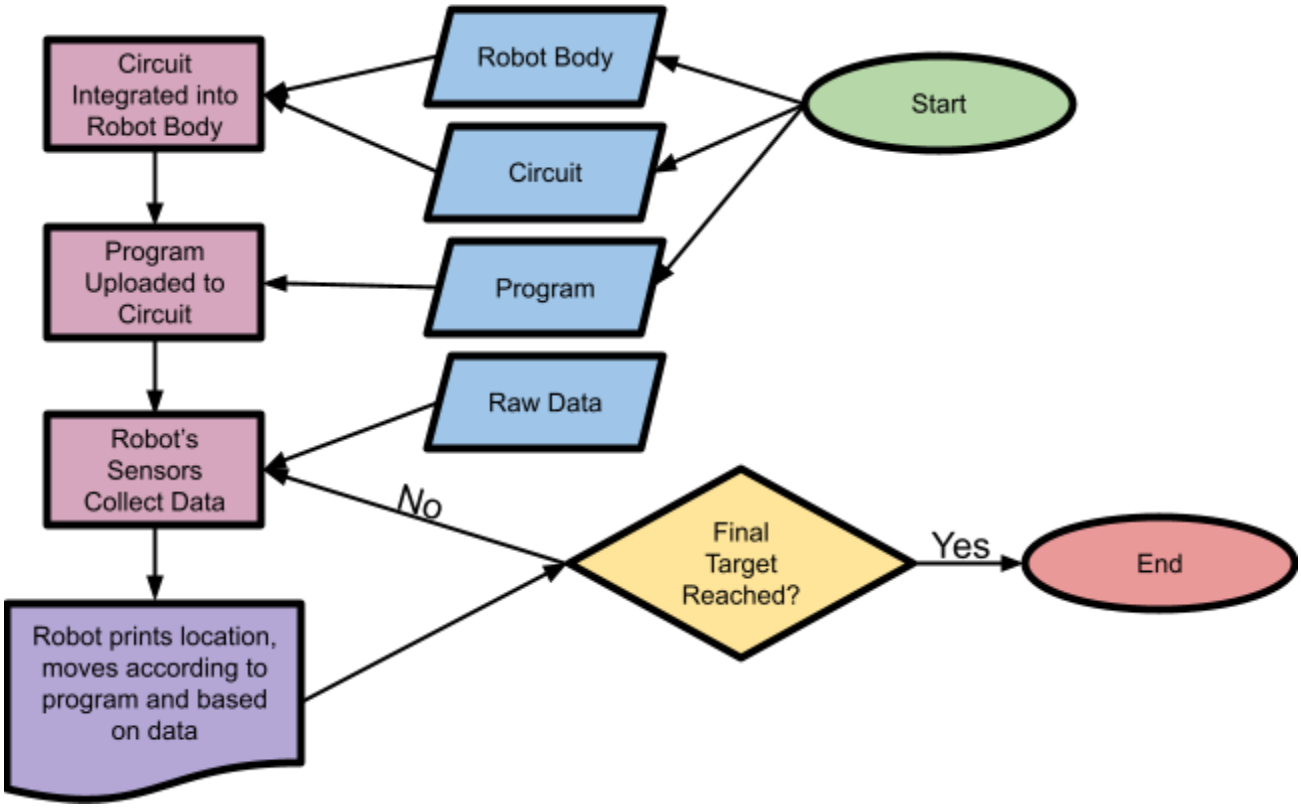
Bill of Materials:

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.
1	ENGR122_ZumoChassis	Includes two silicone tracks, two drive and two idler sprockets	1
2	22s_gps_float_design		1
3	ENGR122_22S_W4_SolidWorksPartDesignBack		1
4	ENGR122_FeeTech-SMC-4CH	Feetech 4 Channel Servo Motor Controller (FT-SMC-4CH)	1
5	ENGR122_WeMosD1	Wi-Fi enabled development Board	1
6	ENGR122_OLED_Display	(0.96 inch) 128x64, SSD1306 Driver I2C	1
7	ENGR122_22S_W4_SolidWorksPartDesignFront		1
8	ENGR122_HCSR04		3
9	Breadboard(proto board)	Dimension: 45.0mm x 34.5mm x 9.5mm	1

System Architecture (See Figure 2)

Overall, the system consisted of three main components: the mechanical design, the electrical circuit, and the software design. The mechanical portion included two 3-D printed models made on SolidWorks. The electrical circuit was made up of three ultrasonic sensors, a WeMos board, a battery, an OLED display, a protoboard, two motors, a motor control, and wires. The software component had codes written for forward, left, and right movement, target and current vector calculations, calculations of the angle between, and the cross product of, the target and current vectors, calculation of distance between current location and target, monitoring when targets were reached, resetting the previous x and y coordinates of the robot, and logic determining which direction to turn, based on sensor data and the cross product of target and current vectors. In combination, these components allow the robot to navigate to four preset targets, pause at each for two seconds, display the robot's current coordinates on an OLED display, and avoid obstacles along the way.

Figure 2



Mechanical Design

The mechanical component of the final design comes in two main pieces: the front and back of the robot's body. These two parts are designed to fit together like puzzle pieces, with holes along the base that align with the holes of the chassis, so that they can be screwed into place. The front piece is equipped with raised holders for two of the three ultrasonic sensors, each tilted at a 30 degree angle away from the central ultrasonic sensor to prevent overlap between sesnors. The central ultrasonic sensor is held in place by two circles that are designed to fit around the extruding portions of the ultrasonic sensor. The front piece is also equipped with a holder for the protoboard, which is a rectangular well located behind the ultrasonic sensor holders. On the back piece of the robot body, there are holders for the motorcontrol (right) and OLED display (left), both of which are raised above the wheels to prevent any interaction with the wheels and to allow maximum visibility. The center of the back piece is occupied by a tall cylindrical holder which is designed to house the GPS float. The height of the cylinder is designed to keep the GPS float vertically centered with respect to the LiDAR sensor, in order to ensure the GPS float is always sensed. At the rear of the back half of the robot body is an envelope-like extrusion. This envelope is designed to hold the WeMos board, with a loose design that makes the WeMos board easy to remove for uploading new code. Finally, the battery and motors are housed within compartments beneath the chassis.

Electrical Circuits and Wiring (See Figure 3)

All parts of the electrical system are connected to the WeMos board which acts as the main centerpiece for the system. The WeMos board can either be powered through a

computer which also serves to upload the code. It can also be powered by the battery pack which can be located inside the chassis of the robot. The battery pack is connected to the VIN and GND ports on the left side of the WeMos board when it is being used using male-to-male wires. The WeMos board also serves to direct the power to all of the other components. These components include the motor control, 3 sensors, and display OLED. In order for all of the components to be connected to the board a circuit board is used to connect several power ports and ground ports to one 5v and GND port on the WeMos board, respectively. There is a line of ports for power and a line of ports for ground with the ground section being above the power. Both are connected to the WeMos board from the circuit board through male-to-male wires. The wheels are powered by the electric motors that are powered and controlled by the motor control that is connected to the WeMos board. The positive and negative ports of the motors are connected to their respective positive and negative “M” on the motor control. Then the “S” port of the motor control is connected to the D0 pin on the WeMos Board which controls the right motor. The other “S” on the motor control corresponds to the left motor and connected to the D2 pin on the WeMos board. This means that the right motor is controlled by the D0 pin and the left motor is controlled by the D2 pin. This can be seen in the code. Both are connected using male-to-female wires. The positive pin for the motor control is connected to the 5v power section of the circuit board. The negative pin is connected to the GND section of the circuit board also using male-to-female wires. For the OLED display, there are 4 ports all of which are being used. They include GND, VCC, SCL, and SDA. The GDN pin is connected to the ground section of the circuit board and the VCC pin is connected to the power section of the circuit board. The SCL and SDA pins of the OLED are connected to the SCL and SDA ports of the WeMos board respectively. For all four of these connections male-to-female wires are used. For each of the 3 sensors, there are 4 pins: VCC, Trig, Echo, and Gnd. For each of the sensors the VCC and GND ports are connected to the power section and ground section respectively. The Trig pin of the front facing sensor is connected to the D8 pin and the Echo of the front sensor is connected to D5 on the WeMos board. For the right sensor the Trig and Echo are connected to D9 and D6 of the WeMos board respectively. For the left sensor the Trig and Echo are connected to D10 and D7 of the WeMos board respectively. This means the front sensor is controlled by D8 and D5, the right controlled by D9 and D7, and the left controlled by D10 and D7. All of the sensors are wired with male-to-female wires.

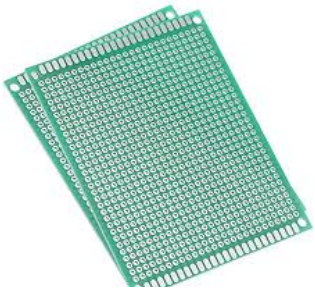
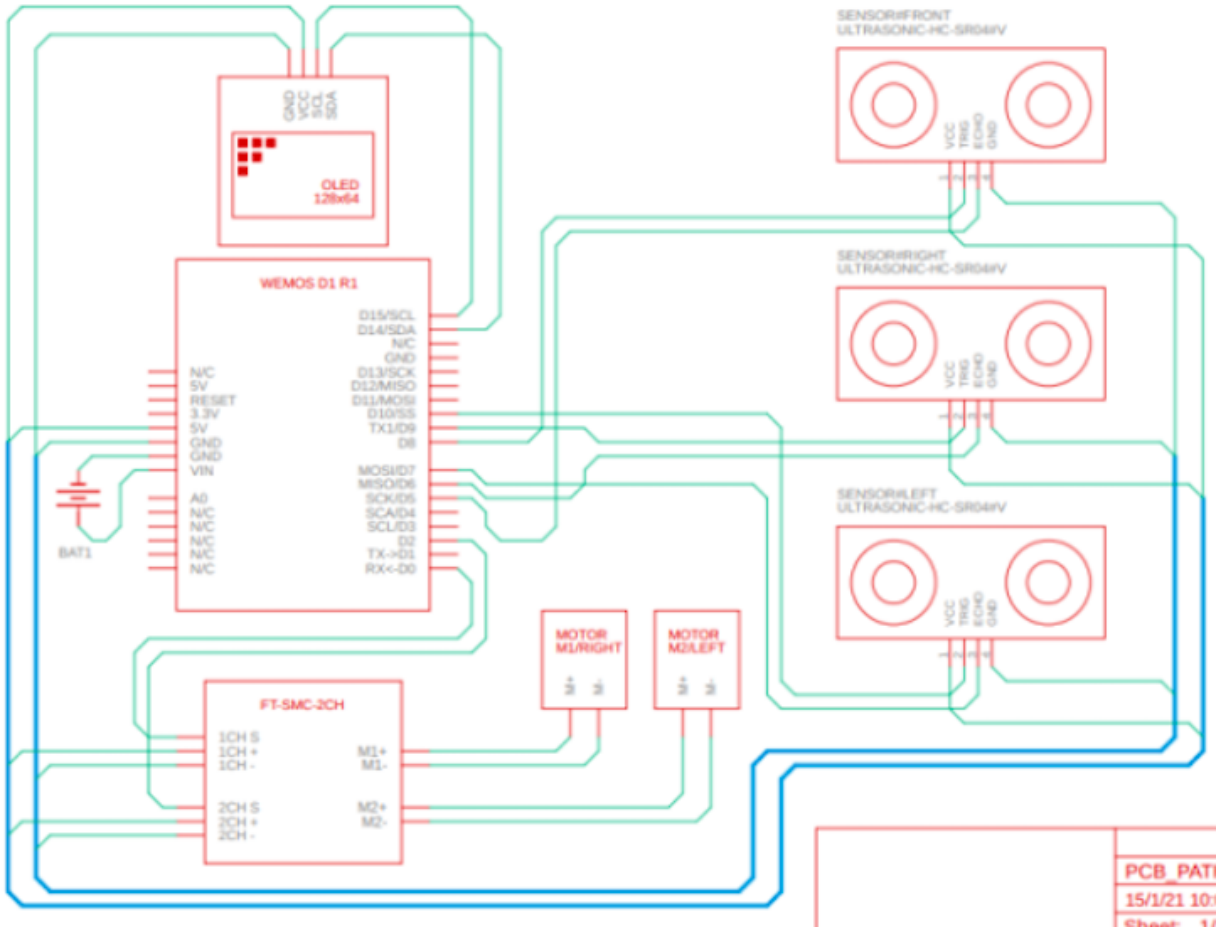


Figure 3: Wiring Diagram



Software Design and Coding

1. Distance Calculation
   This code segment (See Appendix 1) is used for verifying whether or not the robot has reached its target, by calculating the distance between the robot's current location and the target's designated coordinates. If the robot is found to be within ten centimeters of the target, the robot stops for a duration of two seconds, and the index increases, so that the robot will begin measuring distances with relation to the next target.
2. Calculating Current Vector, Target Vector, and Angle Between Vectors
   This code segment (See Appendix 2) is used for determining the vector that the robot is currently following (by subtracting the previous x and y coordinates of the robot from its current x and y coordinates) as well as the target vector (by subtracting the robot's current x and y coordinates from the x and y coordinates of the target). Once these vectors are calculated, it is possible to determine the angle between the vectors by taking the inverse cosine of the dot product of the vectors divided by the product of the magnitudes of the vectors.
3. Cross Product Calculation

This code segment (See Appendix 3) is used for determining the cross product of the current and target vectors. Based on whether this result is a positive or negative number, it is possible to determine which direction the robot needs to turn to approach the target. A positive result signifies the robot needs to turn left to approach the target, while a negative sign indicates a right turn is necessary.

4. Movement Code

This code segment (See Appendix 4) is the primary code used for controlling the robot's motion. The robot first prioritizes going straight, as another section of code (See Appendix 5) focuses solely on turning the robot towards the target. If the robot is unable to go straight due to an obstacle, the robot prioritizes turning towards the target. If the robot is further unable to turn towards the target, the robot will then turn towards away from the target. This section of code also implements code that turns the robot away from obstacles if the robot gets too close.

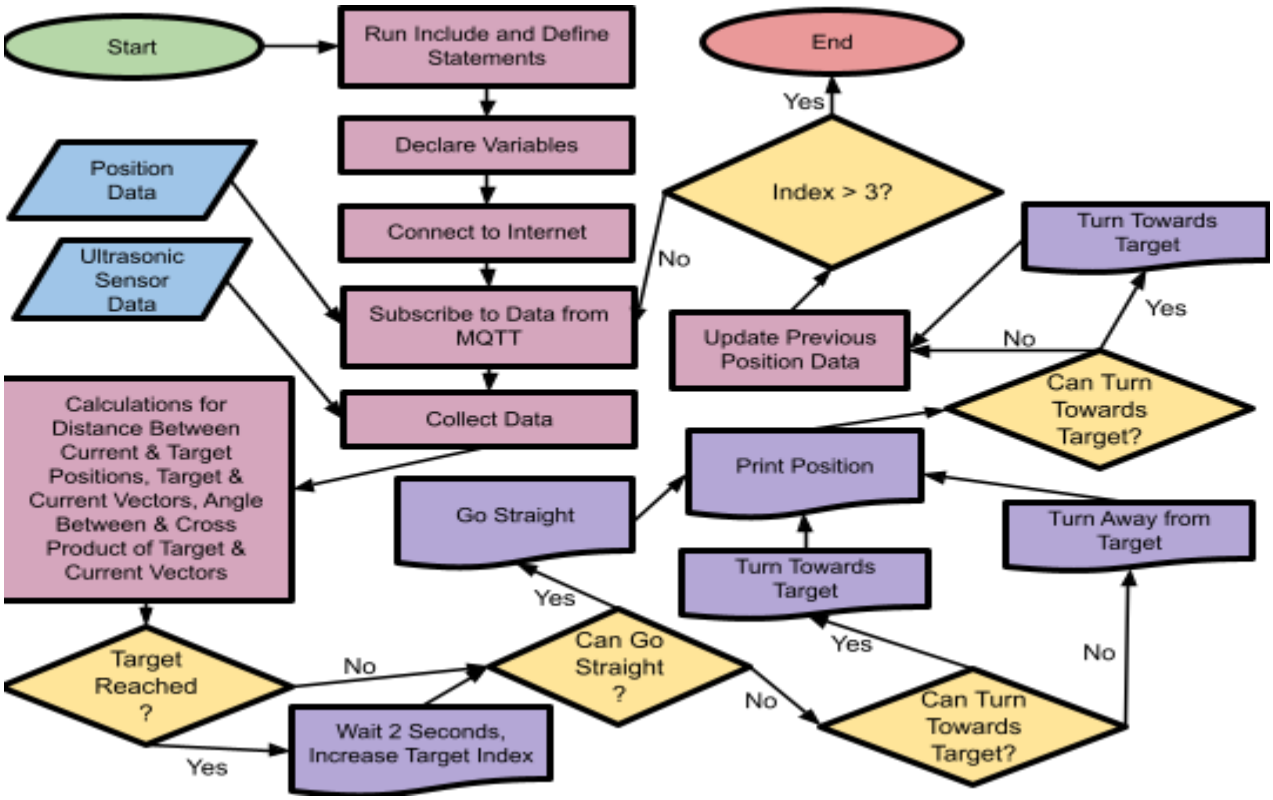
5. Turning Towards Target

This code segment (See Appendix 5) is used for turning the robot towards the target, permitting the sensors do not detect any obstacles. The direction the robot turns is based on the cross product, calculated in another code segment (See Appendix 3).

6. Resetting Previous X and Y

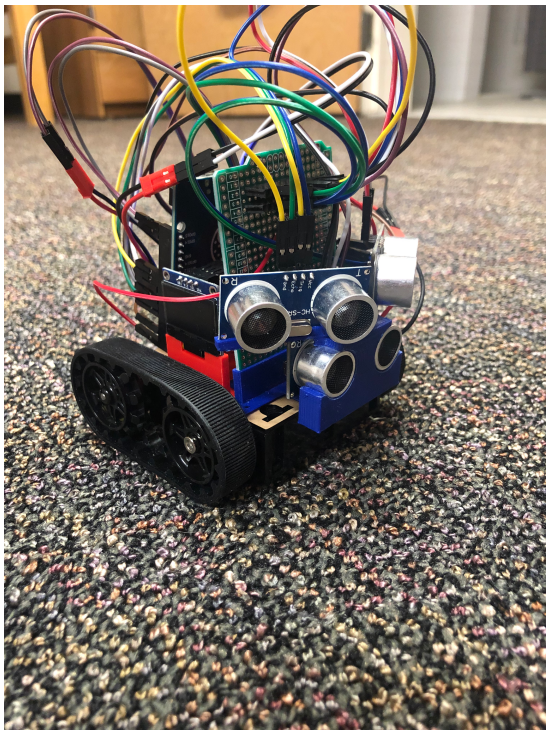
This code segment (See Appendix 6) is used for providing new previous coordinates so that the current and target vectors, as well as the calculations dependent on those vectors, will remain accurate. The values are replaced with the current coordinates, once the coordinates have been used in calculations.

Figure 4: Block Diagram of Code



Assembly and Prototyping (See Figure 1:4 and Figure 1:5)

Assembly of the robot starts with connecting the motors, wheels, and treads to the chassis. Next, the front and back robot body parts are attached with screws to the chassis, aligning the holes so that no holes are covered. After, the central ultrasonic sensor is placed into its holder before the other parts, as once the protoboard is put into its holder this sensor is much more difficult to remove. The central ultrasonic sensor should be positioned facing forwards with its connecting prongs positioned upwards. The remaining parts can be placed into their locations in any order. The OLED display should be placed on the left side of the robot, with its screen facing outward. The motor control is placed on the right side of the robot, such that the side with six connecting prongs is facing forwards, and the side numbered "1" is on top. The WeMos Board should be placed with the connecting ports facing the rear of the robot, and the micro-USB connection facing upwards. The protoboard should be placed such that the portion of the board where the wires are connected is on top, facing the front of the robot. The remaining ultrasonic sensors should be placed facing approximately the same direction as the central ultrasonic sensor, angled slightly to either side.



Performance

After finalizing the code and assembling the robot body, when put in the arena to complete the course, the robot didn't perform very well. The robot wasn't able to complete any of the targets. The robot was able to perform basic operations as the code allowed the robot to run but the wiring of the robot didn't allow for it to complete any of the targets. The robot mainly had issues in terms of obstacle avoidance as it often ran into walls, or would spin in place after detecting an obstacle. The robot had trouble redirecting itself as it would often lose direction and repeat the same cycle of running and turning from a wall. Before

putting the robot in the arena, we tested the code several times, to figure out what the problems were in order to adjust the code. For example, when we noticed the robot was moving too fast and missing the target, we added a longer delay in the code to fix the issue. With each adjustment made to the code or when considering new wiring arrangements, we uploaded the code onto the robot to ensure the code was functional and to make sure any new changes didn't make us decline in terms of progressing the robot. For example, when testing out a new wiring arrangement we found that it wasn't beneficial and made the robot perform worse than the original configuration.

Concluding Remarks

The project called for the creation of a robot that could move on its own, print its coordinates regarding the testing arena. It was supposed to efficiently avoid obstacles and move to 4 different target locations. Once reaching a target location it was meant to stop for a certain amount of time. In the early stages we were able to successfully create a robot that could move on the track. Our initial design consisted of a basic robot body that could hold sensors and the other components were only connected through the wiring and would just hang loosely around the robot. Our initial prototype was able to successfully navigate itself through the track and also was able to avoid collisions into the wall by moving either closer to it or farther away depending on where it needed to be. It also would turn 90 degrees when approaching a wall from the front to avoid collisions in the front. The final requirements of creating a system that could avoid obstacles and move to desired target locations was not successfully completed, but the robot body was improved upon. The final robot body used two different sections that were able to hold all of the robotic components. There were no longer any loose or hanging parts. Also, the breadboard was replaced with a circuit board as required. This circuit board was custom soldered to connect several power and ground pins to two single ports on the WeMos board. The final design implemented 3 sensors. The reason for our inability to complete the project was due to the software design, as the robot was never coded properly to reach the target destinations. Also, there was a wiring issue that held us back on testing our code. In the end we have a robot with a functioning OLED display screen that is able to track its coordinates and move in the testing arena. The problem with our wiring was with the motor control. The problem was that the motor control was being powered twice since we thought that each motor should have separate power connections. Another electrical issue was with the sensors that had incorrect trig and echo pins. With these issues avoided we would have had more time to come up with a working software design. Next time this project is attempted the motor control should ont be connected to the power pins of the WeMos board once.

Appendix

1. Distance Calculation

```
distance = sqrt(sq(yt[j]-y)+sq(xt[j]-x));
 if(distance <= 100){
 delay(2000);
 i++;
 j = i;
 }
```

2. Calculating Current Vector, Target Vector, and Angle Between Vectors

```
 double vxCurrent;
 double vyCurrent;
 double vxTarget;
 double vyTarget;
 vxCurrent = x - previous_x;
 vyCurrent = y - previous_y;
 vxTarget = xt[j] - x;
 vyTarget = yt[j] - y;
 double param, result;
 param = ((vxCurrent*vxTarget) + (vyCurrent*vyTarget)) / ((sqrt(vxCurrent*vxCurrent +
vyCurrent*vyCurrent))*(sqrt(vxTarget*vxTarget + vyTarget*vyTarget)));
 result = acos (param) * 180.0 / PI;
```

3. Cross Product Calculation

```
 double crossProduct;
 crossProduct = (vxCurrent*vyTarget) - (vyCurrent*vxTarget);
```

4. Movement Code

```
if(distance_front > 10 && distance_right > 3 && distance_left > 3){
 motorR.write(180); // Request to generate the PWM signal
 motorL.write(0);
 }
else if(distance_front > 10 && distance_right <= 3){
 motorR.write(0);
 motorL.write(0);
}
else if(distance_front > 10 && distance_left <=3){
 motorR.write(180);
 motorL.write(180);
}
else if(distance_front < 10 && distance_left > 10 && crossProduct > 0){
 motorR.write(180);
 motorL.write(180);
}
else if(distance_front < 10 && distance_right > 10 && crossProduct < 0){
```

```
  motorR.write(0);
  motorL.write(0);
}
else if(distance_front < 10 && distance_left > 10){
  motorR.write(180);
  motorL.write(180);
}
else if(distance_front < 10 && distance_right > 10){
  motorR.write(0);
  motorL.write(0);
}
```

5. Turning Towards Target

```
if ((crossProduct > 0) && (distance_front > 6 && distance_left > 6)){
  motorR.write(180);
  motorL.write(180);
}
if ((crossProduct < 0) && (distance_front > 6 && distance_right > 6)){
  motorR.write(0);
  motorL.write(0);
}
```

6. Resetting Previous X and Y

```
  previous_x = x;
  previous_y = y;
```