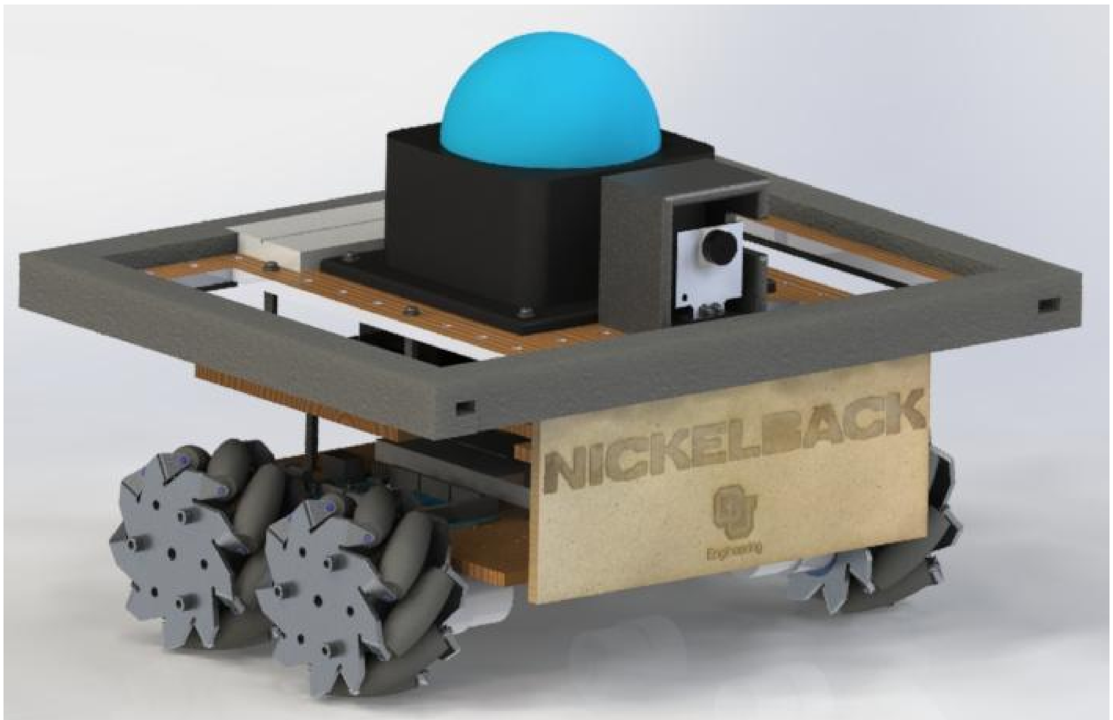
MCEN 4115/5115: Mechatronics and Robotics I
Project Report

Autonomous Rocket League Robot



Team Nickelback

Nic Garzione, Kenneth Colanto, Emily Levis, Omar Gutierrez Cardoza, Justin Varghese

December 9, 2021

Table of Contents

1.0 Introduction

As the use of robotics continues to grow in industry and research, so has the need for autonomous robotics that can complete a wide range of tasks. One area of growing interest is the ability for a robot to track objects autonomously while navigating in an environment and avoiding obstacles. This has a variety of applications such as performing rescue operations in hazardous environments, delivering materials to remote areas, and assisting in military services. In this project, we begin to explore this area of research by designing a robot tasked with competing against other robots in a real life version of the video game Rocket League, in which players control cars in a game similar to soccer. The key challenge is to detect the ball and maneuver it into the opponent's goal net, either by pushing it or kicking it forward, while also avoiding the opponent robot.

The robot can detect its surrounding environment in a variety of ways, including image processing, radio signal, and/or sonic detection. This task allows for the team to test different potential approaches and compare them to each other through the competition. This report outlines our approach, design process, manufacturing and assembly methods, algorithm used in navigations, and functionality for our autonomous robot.

2.0 Mechanical Design

The robot was designed to perform its two main tasks of scoring goals and stopping an opposing robot from scoring. The attributes that allow the robot to perform these two tasks are its agility and its ability to dribble effectively. This section goes over the reasons behind the mechanical designs of the robot that allow it to perform its tasks and showcases how each part was manufactured along with a bill of materials (BOM).

2.1 Chassis Design

The robot chassis was designed with the intent to maintain simplicity while allowing the robot to store all necessary electrical components and keeping them safe from possible damage during the game.

The overall design of the robot began on Chassis Level 1, pictured in figure 1. A simple square design allowed for easy installation of the motor/bracket/wheel sub assembly. The space in the middle of the chassis was designated for initially storing our electrical equipment, specifically the Arduino Mega 2560, Raspberry Pi, Pixy camera, and battery. A standardized hole pattern that is used on each of the 3 chassis levels is shown in figure 1 which allowed for freedom of placement for all components throughout the design. This simple base chassis allowed for initial testing of the software and to test independent system functionality.

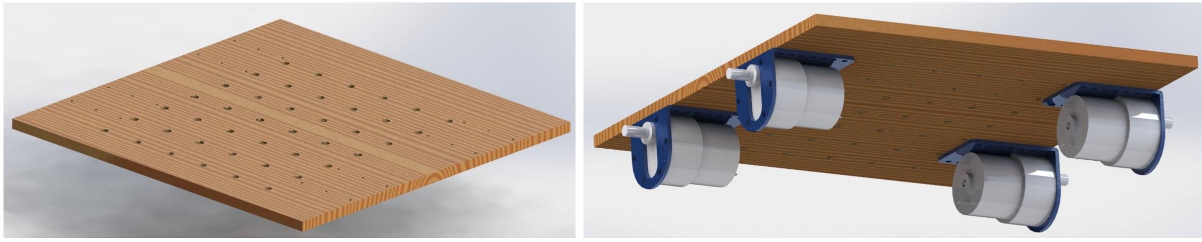
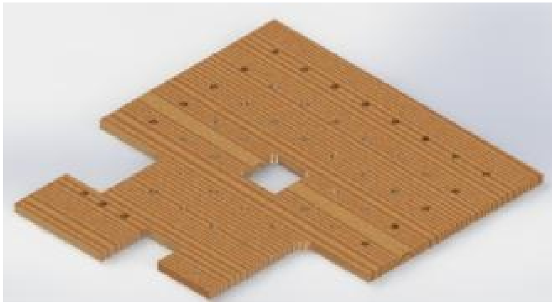


Figure 1: Chassis level 1 (left) and motor/ bracket sub assemblies attached to chassis level 1 (right).

To allow the robot to move in any direction rapidly on a 2-D plane with good agility, mecanum wheels were determined to be the best and easiest solution. Mecanum wheels provide this dexterity by having certain wheels rotate in the opposite direction and at different speeds than others by simple voltage changes to the motors. Figure 2 shows the two types of wheels that allow for these movements.

Figure 2: Image of mecanum wheels showing the two different configurations of the rollers.

The design requirements stated that the robot must not exceed a rectangular footprint of 12” x 14” and a height of 11”. The need for more storage of electrical components arose and the decision to add chassis layers up to 11” high was made. Chassis layers 1 and 2 are connected by sets of 3 M3 standoffs that are 15mm tall and connect to the top and bottom of each layer with M3 screws and washers to increase surface area for fastening. Initially, chassis layer 2 was designed to mount our ‘kicker’ that would have been used to send the ball forward by attaching a small LEGO wheel to a high speed motor. When this wheel is rotated at a high speed and makes contact with the ball, it would ‘kick’ the ball. However, the plan to use this motor was eventually scrubbed because the ball proved to be too heavy for our ‘kicker’ since it could not substantially move the ball. The design for chassis level 2 was kept since the area for the kicker motor and wheel did not affect the rest of the robot. Chassis level 2 was also designed to house the Raspberry Pi and its power supply so a square cut in the middle was made for wires to reach the breadboard on chassis layer 1. Figure 3 shows an image of chassis level 2 by itself and an image of chassis layer 2 connected to chassis layer 1 with standoffs.



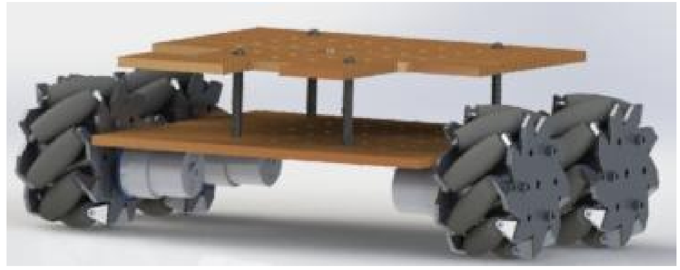


Figure 3: Chassis layer 2 (left) and chassis layer 2 connected to chassis layer 1 with wheel/motor/bracket assembly.

Another design requirement was for the robot to have a handle or handles so team members and referees could pick the robot up without causing damage. The robot also needed to have a bumper that would be the first point of contact to other robots. This bumper had to sit in the range of 6” to 8” from the ground. Lastly, the robot needed to house a cup or ring to hold a green or blue 3.75” diameter ID ball in the geometric center of the robot for the pseudo-GPS system to track the robot on the field. Chassis layer 3 ensures that all of these requirements are met.

Chassis layer 3 acts as the bumper for the robot with dimensions of 11” x 13”. A ½” wide foam layer with a ¼” slot was attached to the perimeter of this layer for energy dissipation from impacts with the opposing robot. The foam adds an inch to each side of chassis layer 3, bringing the overall dimensions to 12” x 14”, the maximum footprint size allowed.

Chassis layer 3 also has square slot cutouts to form handles on either side of the robot. Placing these slots here allows for even weight distribution when lifting. An image of chassis layer 3 can be seen in figure 4.

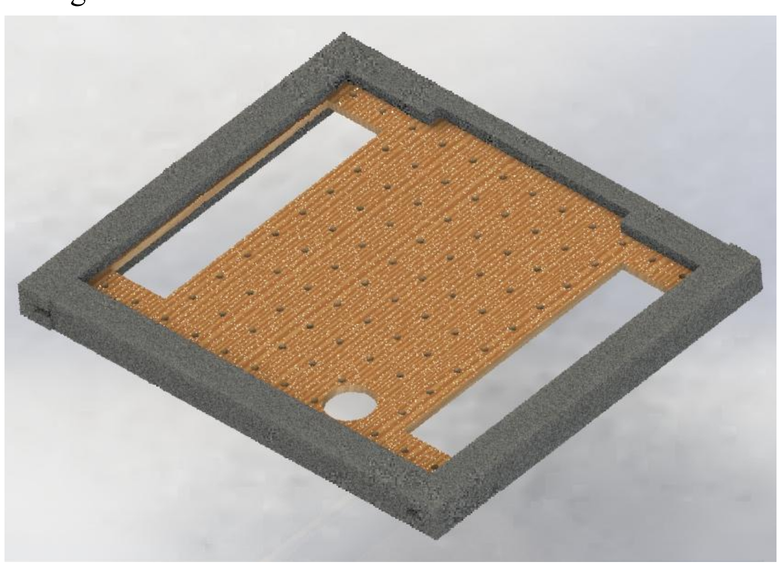


Figure 4: Chassis layer 3 with foam bumper

The cup for the identification ball is also housed on chassis layer 3 along with the Pixy camera and its covering/mount. A breadboard circuit responsible for changing the ‘mode’ of the robot to green or blue and resetting the robot is also housed on this layer. Figure 5 shows all of these components mounted on chassis layer 3.

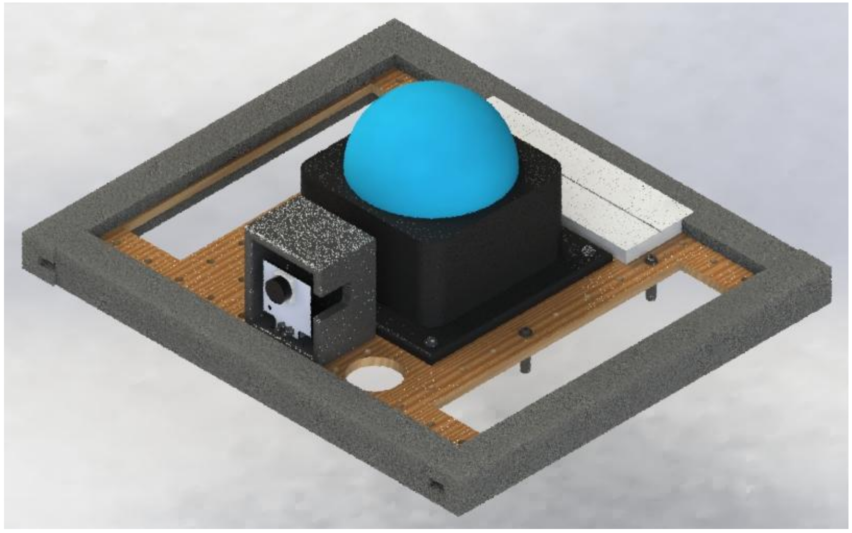


Figure 5: Chassis layer 3 with the breadboard, ID ball cup (with ball), Pixy camera and its covering/mount.

Chassis layer 3 is attached to chassis layer 2 with sets of 2 M3 standoffs that are 20mm tall, creating a separation of 40mm between chassis layers 2 and 3. The overall height of the robot from the ground to the top of the ID ball is 10.96” which is just under the maximum limit of 11”.

The design of chassis layers 1 and 2 were partly focused around the placement of the electrical components. Chassis layer 1 is responsible for housing the Arduino Mega 2560, battery, and the main breadboard circuit. Chassis layer 2 is responsible for housing the Raspberry Pi and its power supply as well as another Arduino for the radio. The radio is attached to the side of the ID ball cup on chassis layer 3 and has wires going to the breadboard on chassis layer 3. Wires for the radio wrap around the back of the robot from chassis layer 3 to the Arduino housed on chassis layer 2. All wires and the radio were not modeled in Solidworks. Figure 6 shows the electrical components housed on their respective chassis level.

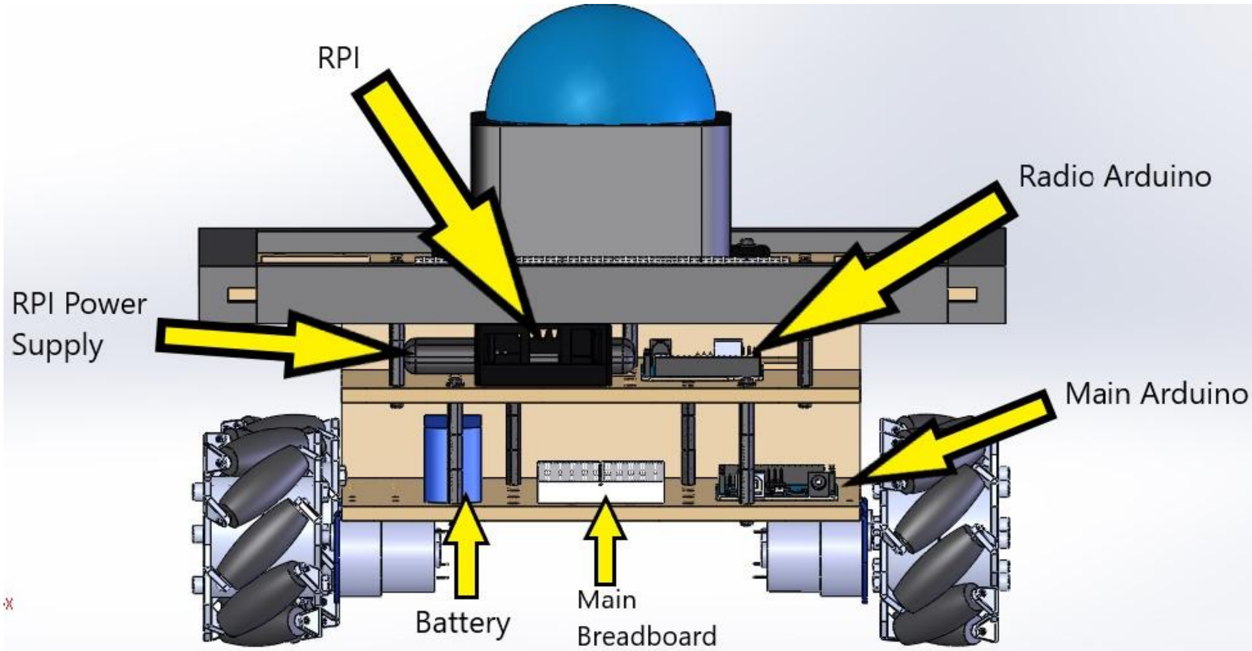


Figure 6: Rear view of assembled robot with arrows indicating locations of electrical components.

As testing progressed with our robot locating the ball and driving into it, the robot continued to get stuck on the ball when it made contact with it because the ball was getting stuck in the openings between the chassis layers. As a result, the robot was climbing onto the ball and the front wheels were being lifted into the air. To fix this issue, a 9" x 4" front plate was adhered to the front of the robot, covering the openings between the chassis layers. An image of the robot before and after the front plate was added is shown in figure 7.

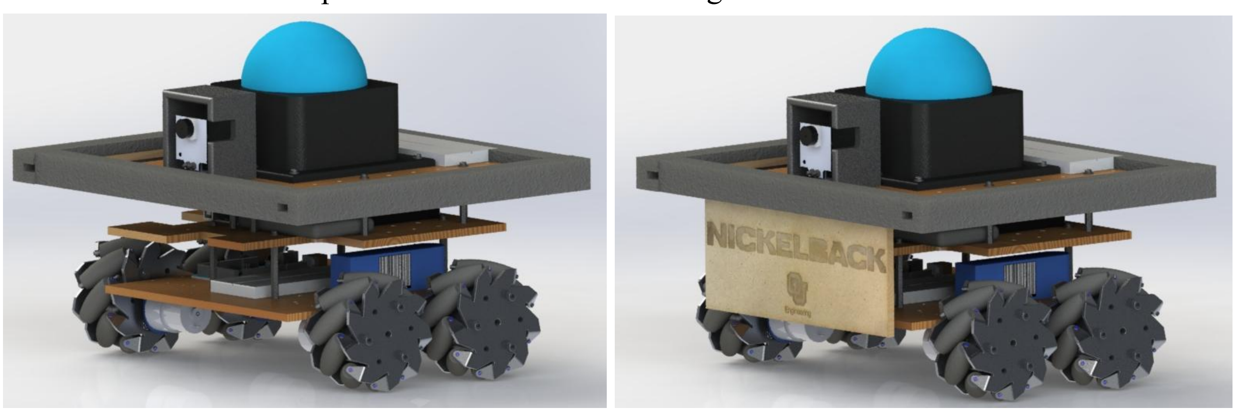


Figure 7: Robot without front plate (left) and robot with front plate (right).

With the addition of this front plate, the robot is able to successfully 'dribble' the ball effectively without getting stuck. An image of the actual robot is pictured below in figure 8.

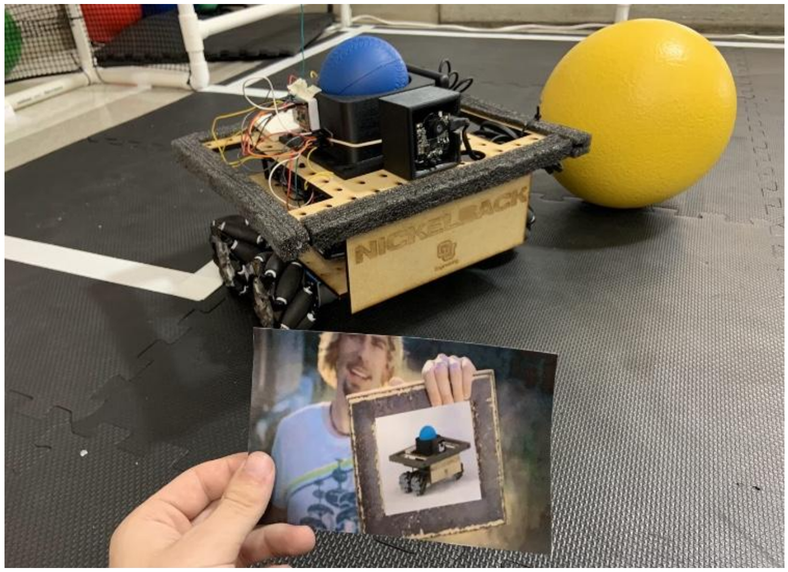


Figure 8: Image of actual robot

2.2 Manufacturing

Minimal materials were used in the overall design of the robot. For the 3 chassis layers and front plate, ¼" MDF board was laser cut from 2' X 2' X ¼" stock. Solidworks was used to design the chassis layers in part files and then they were converted into .dwg files so they could be compatible with the Epilog Legend laser cutter.

Other parts that were manufactured were the ID ball cup and the Pixy camera housing/mount. Both of these components were 3D printed from black PLA filament on the TAZ Workhorse 3D printer. These two 3D printed components were fastened to chassis layer 3 via M3 screws into standoffs that were holding chassis layer 3 from chassis layer 2. L-brackets for the Pixy came with the camera in its packaging, so the Pixy camera housing/mount were designed around using these brackets. With the Pixy standing on its end with the camera facing forward, 5mm M3 screws with nuts were used to fasten the pixy to the L-brackets. These brackets were fastened to the housing/mount via M3 screws that went through chassis layer 3 and the bottom surface of the housing/mount. Figure 9 shows chassis layer 3 isolated from the robot with the two 3D printed components mounted to it.

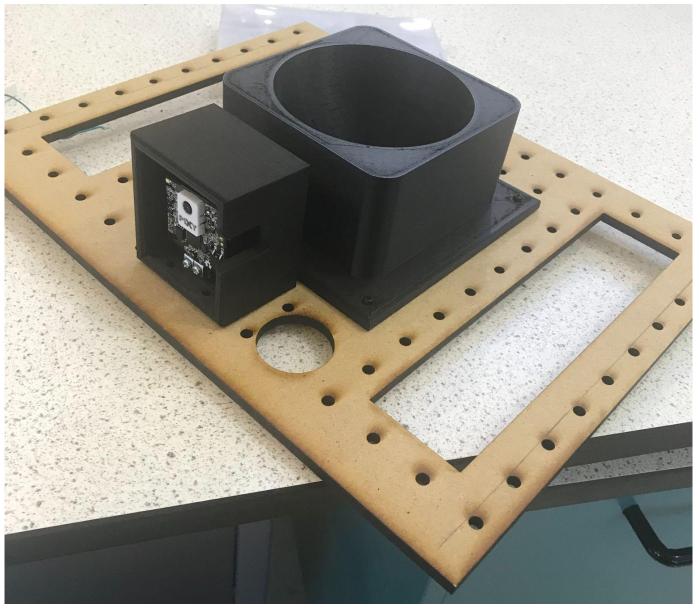


Figure 9: Image of 3D printed components mounted to the isolated chassis layer 3

Lastly, slotted foam bumpers were also manufactured for the perimeter of chassis layer 3. Three “12 X 12” X 1” foam plates were used as the stock material for the foam bumpers. ¼” thick slots were cut into the face of the foam stock. With a ¼” left on either side of the slot, the foam bumper was cut out with each bumper being 12” long. Each bumper was glued onto the 4 sides of chassis layer 3, where the edge of the layer fit into the slot cut out of each bumper.

2.3 Bill Of Materials:

Borrowed from class/university	Quantity	
Motors	4	
Motor Mounts	4	
LiPo Battery	1	
Pixy2 Camera		
Borrowed from Personal Equipment	Quantity	
GY-521 MPU	1	
Motor drivers	3	
Arduino Mega	2	
Raspberry Pi	1	
Purchased	Quantity	Total Cost ($)
Battery Pack for Raspberry Pi	1	24.99
MDF Board (1/4") 2'x4'	2	19.99
Polyethylene Foam	1	7.99
Velcro Tape Roll	1	6.97
600PCS M3 Nuts and Bolts	1	15.99
Lego 8PC Wheel and Tire Set	1	12.99
Mecanum Wheel Kit + 6mm Motor	1	36.99
320PC Standoff Screw Set	1	11.88

3.0 Electronics System

3.1 Raspberry Pi Diagram

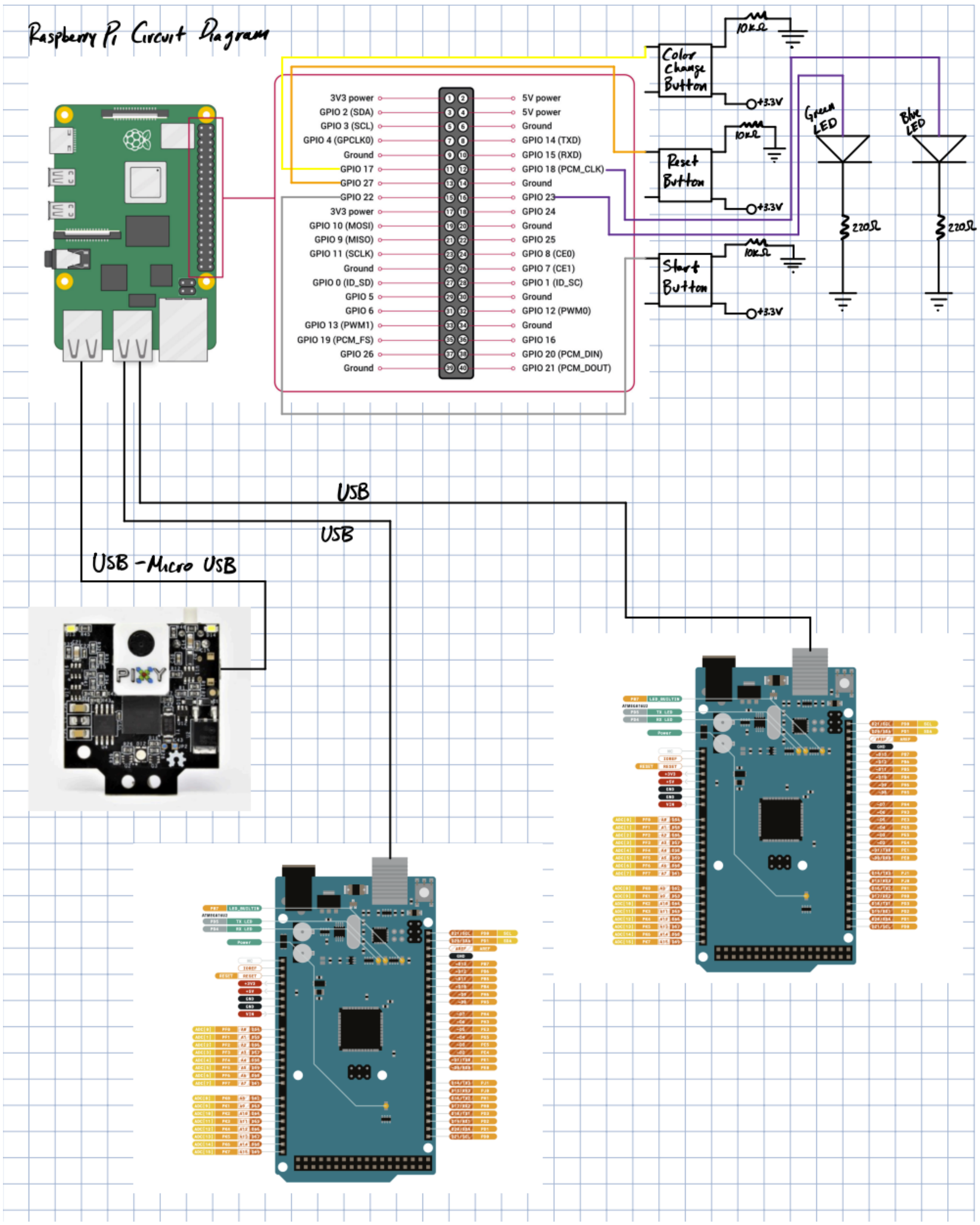


Figure 10: Raspberry Pi/Pixy/Arduino/Buttons/LEDs Circuit Diagram

3.2 Arduino Mega #1 Diagram

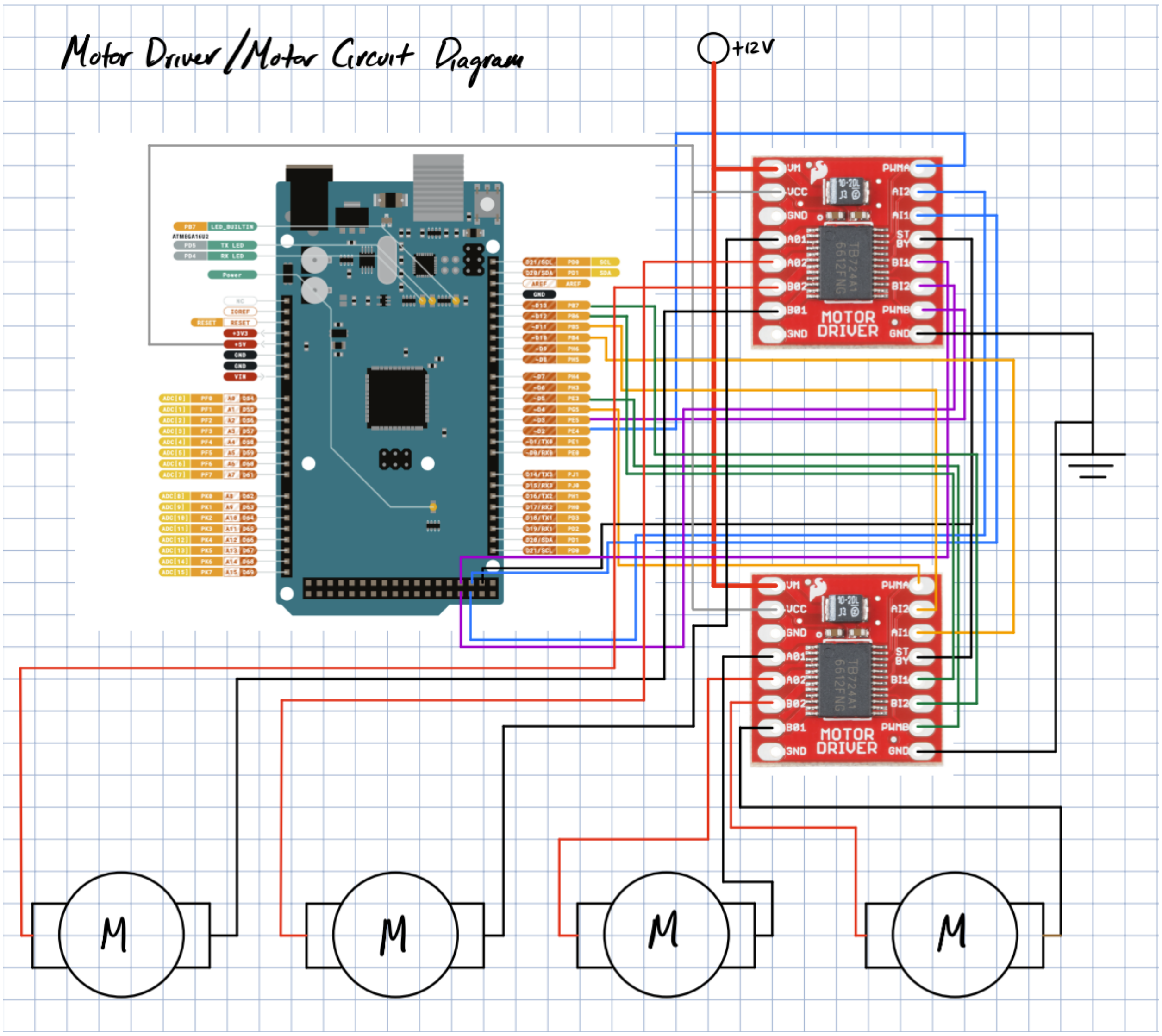


Figure 11: Motor Driver/Motors Circuit Diagram

3.3 Arduino Mega #2 Diagram

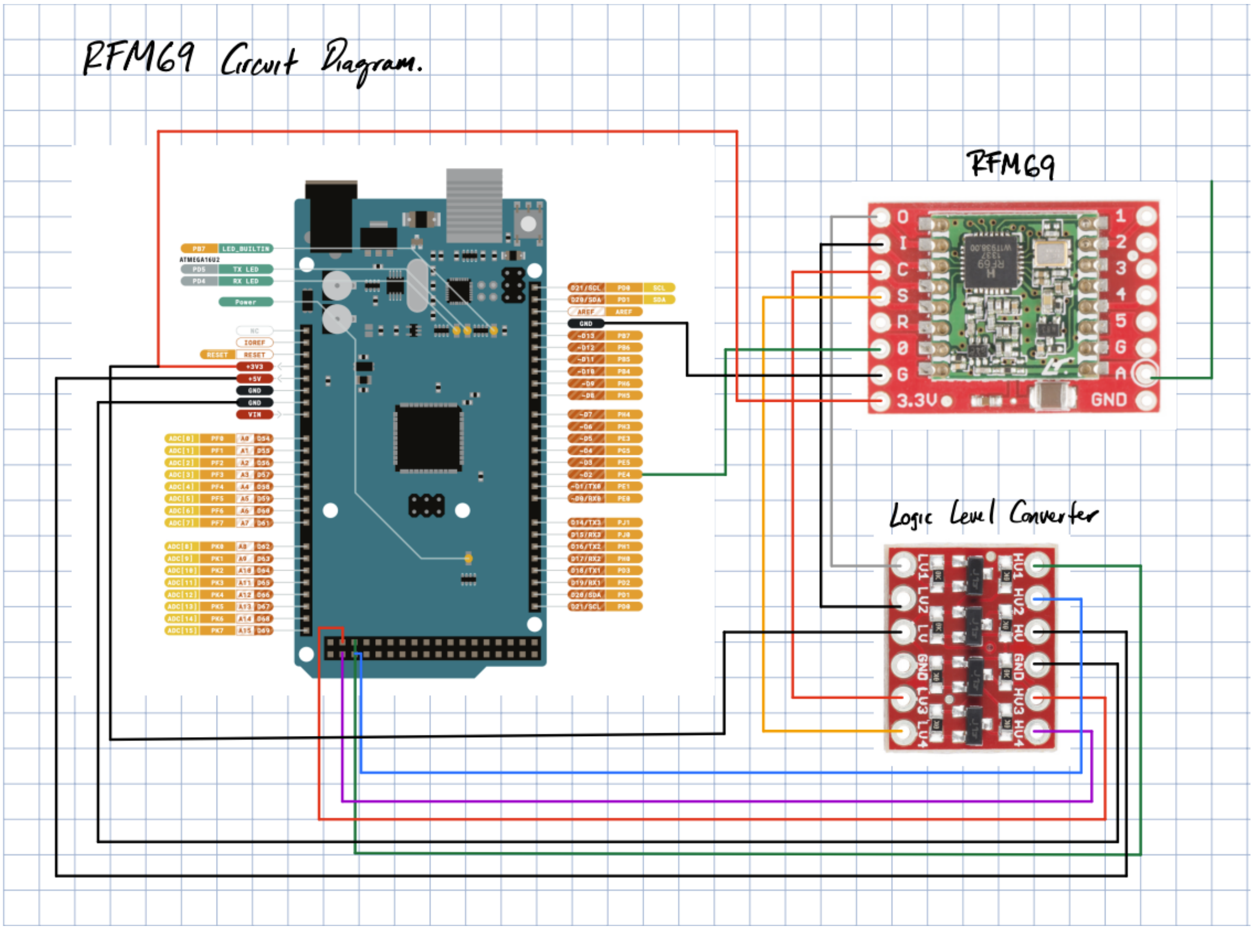


Figure 11: RFM69 Radio/Logic Level Converter Circuit Diagram

3.3 Arduino Mega #1 Diagram Continued

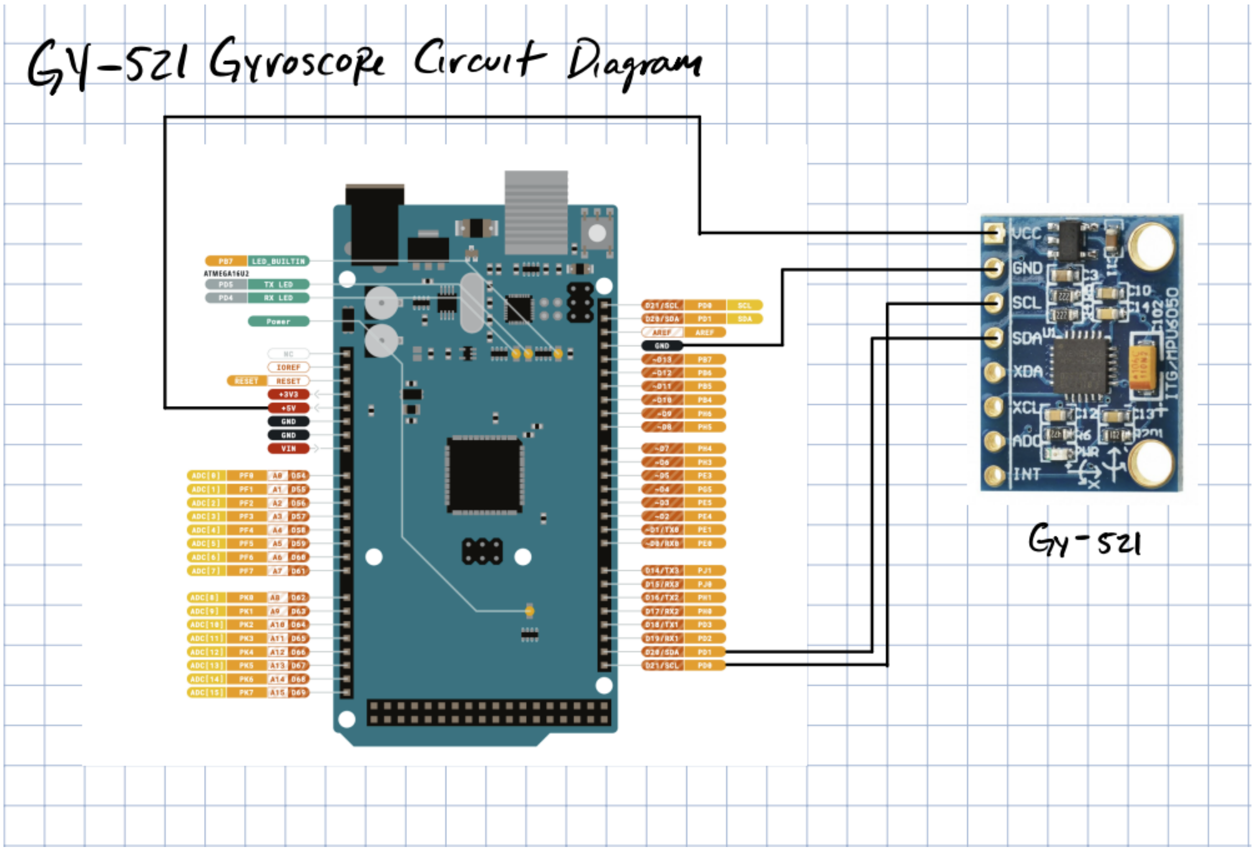
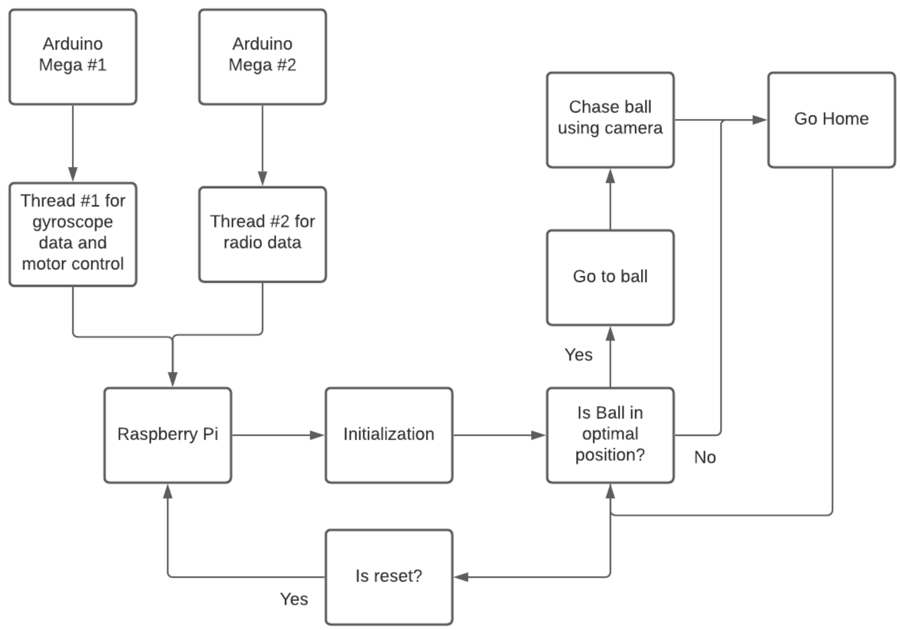


Figure 12: GY-521 Gyroscope Circuit Diagram

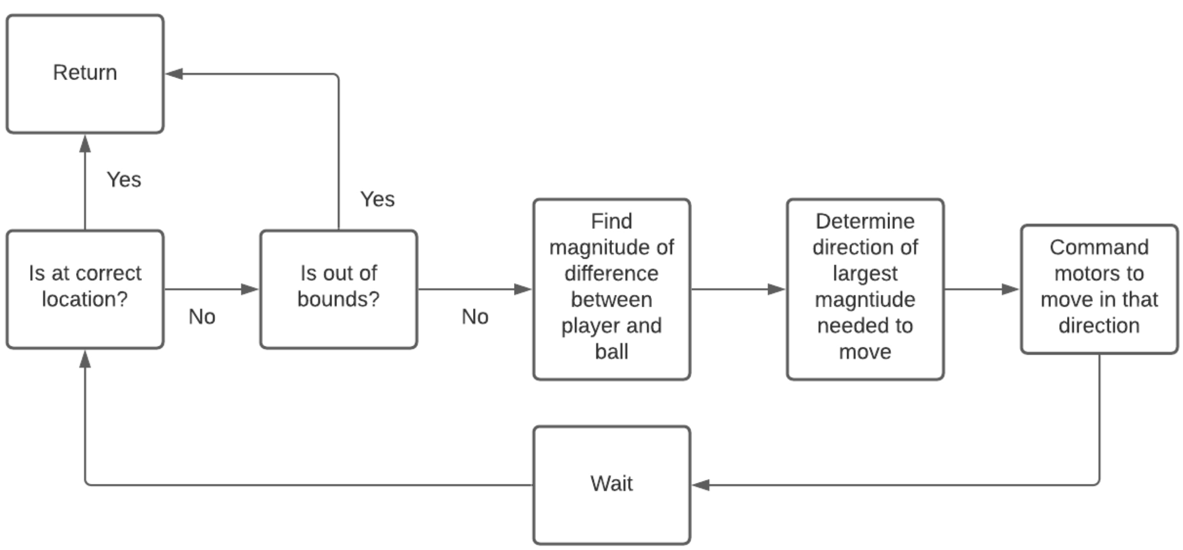
4.0 Software

4.1 Raspberry Pi Code

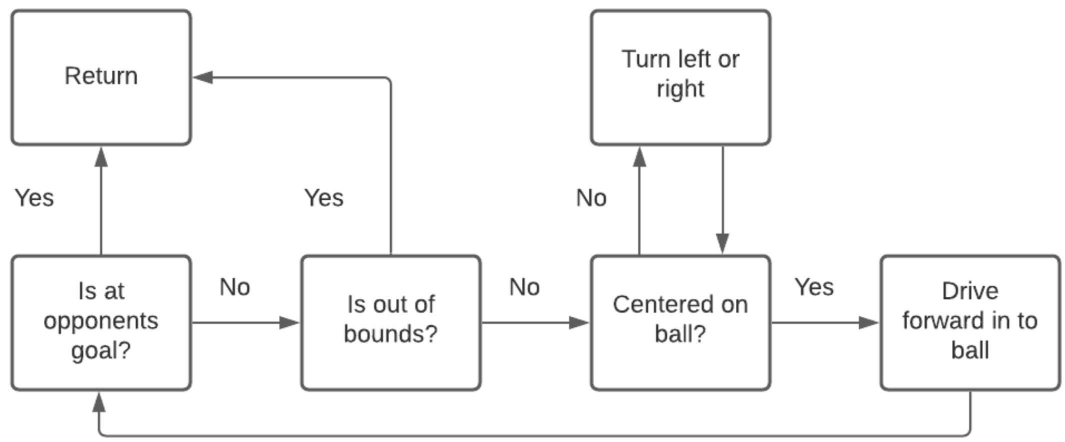
High level overview:



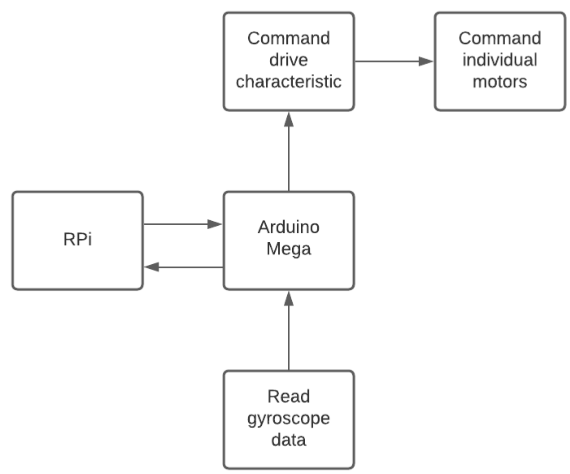
Path Planning:



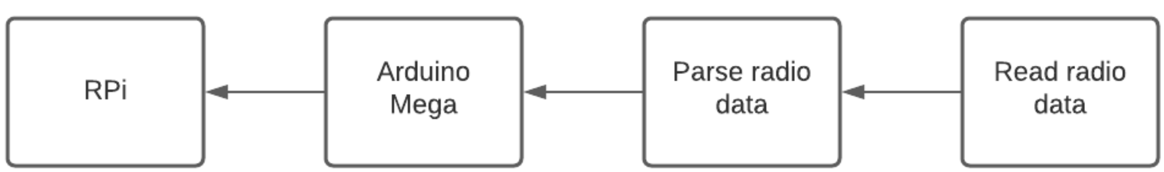
Using Camera to Chase Ball:



4.2 Arduino Mega #1 Code



4.3 Arduino Mega #2 Code



5.0 Lessons Learned

The final competition gave a lot of insight into what strategies and components benefited the performance of the robot in completing the tasks. Robots that focused on the camera detection for finding and chasing the ball performed the best overall. However, they also struggled with aiming towards the goal and would often become stuck ramming the ball against the borders of the field. The robot also needed a slight delay as it turned to actually process that it saw the ball, and robots that did not have this delay consistently missed the ball completely.

Our own robot focused on a path finding algorithm using the coordinates of the objects on the field. When it was close enough to the ball, it would switch to the camera and use image processing to locate the ball. Our robot performed well when the ball was directly between it and the opponents goal, but would get lost otherwise. This may be because our gyroscope wouldn't calibrate correctly, resulting in the robot being unable to detect its location accurately. Additionally, when it came to our image processing, our robot had difficulty fully distinguishing the ball from the surrounding. Moving forward, the robot may have benefited from the use of machine learning to identify the ball from its shape as well as color. A more accurate image processing program, working simultaneously with the path finding algorithm, would allow the robot to maneuver the ball and aim it towards the goal more precisely.

Appendix

GitHub Repository: https://github.com/nicolasgarzione/rocket-league-robot