ECE-118

Professor Gabriel Hugh Elkaim

Final Lab Report:

June 14, 2024

Authors:

Ana Elizabeth Villa (Liz)

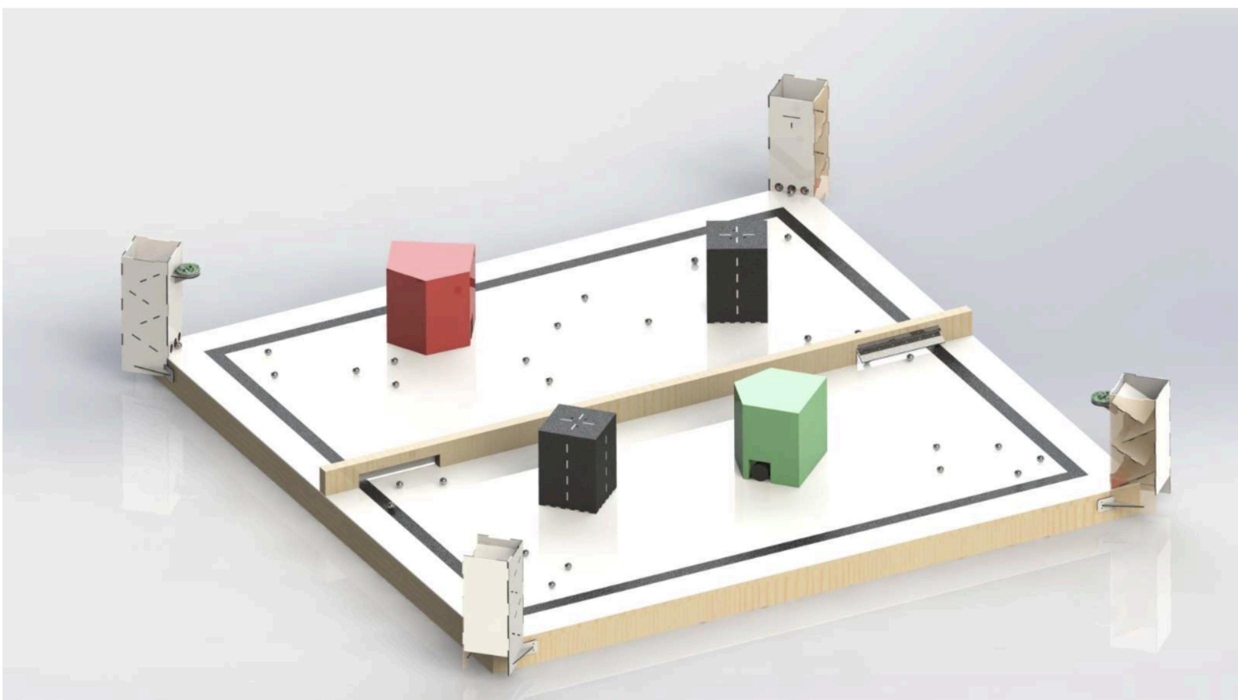
Rafael Delwart

Introduction

Project Overview:

In the final lab for Mechatronics the task assigned was to create an autonomous robot that will navigate the game field, locate and trap randomly rolling 25mm chrome balls, and store them within your robot or push them onto your opponent's field, cleaning up as much as possible in the 2-minute round. The minimum specifications to complete this task is to collect 30 balls total including the ones deposited and there must be at least 2 balls deposited to successfully complete the task.

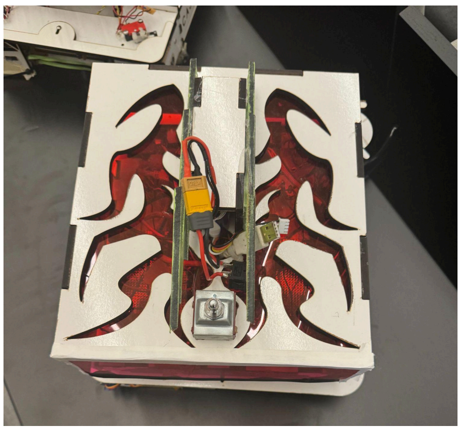
The field has two towers per side with a beacon detector located at the tower directly across from the trapdoor where you must deposit your balls. Above the trapdoor there is a track wire where teams could utilize the signal to ensure they line up with the trapdoor everytime. The field is outlined with black tape and there is one object placed randomly on the field which must be avoided.
Below is a diagram of the field:

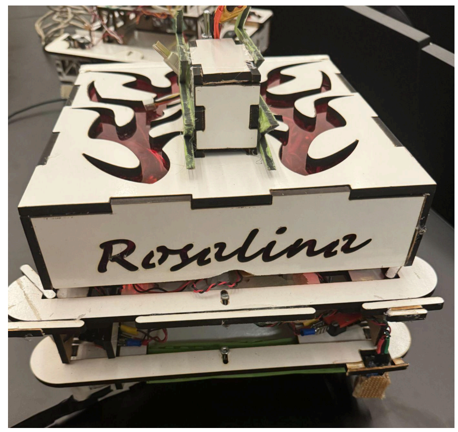


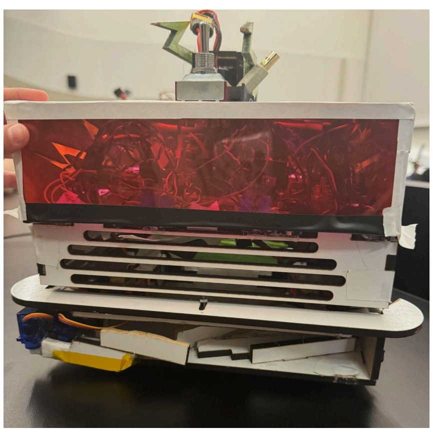
Logic for our Design:

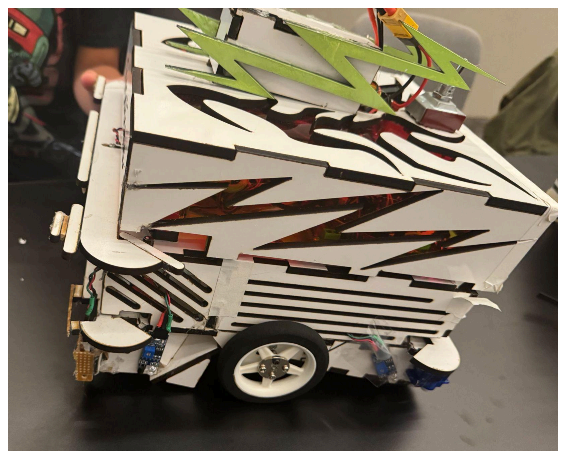
The approach taken to complete this was to use only tape sensors and bumpers to navigate the field. There are 6 bumpers and 5 tape sensors, the bot design has two layers , a wall detection layer(4 bumpers) with tape sensors (5 tape sensors), and an object detection layer(2 bumpers). For the navigation/ software of the bot it was decided to have it drive forward and from there there are only 3 possible events that could occur. One is that it hits the wall first which would then send the bot into a wall following state where it would follow the wall going counter-clockwise since the trapdoor is on the right side. Second it could hit the tape first which would follow the same navigation logic as the wall but using tape sensors. Third, it could hit the object first which would just send it to a simple object avoidance state. No matter where it started it would eventually end up at the trap door with the front two bumpers hitting the trapdoor first. Once it did it would be sent into a backup, 180 dance, backup again, open the trap door and deposit the ball's state. From there it would travel around the field again until it hit the trap door again to deposit. The strategy for collecting the balls was hoping to collect enough everytime we would try and look for the trapdoor again.

Final Robot







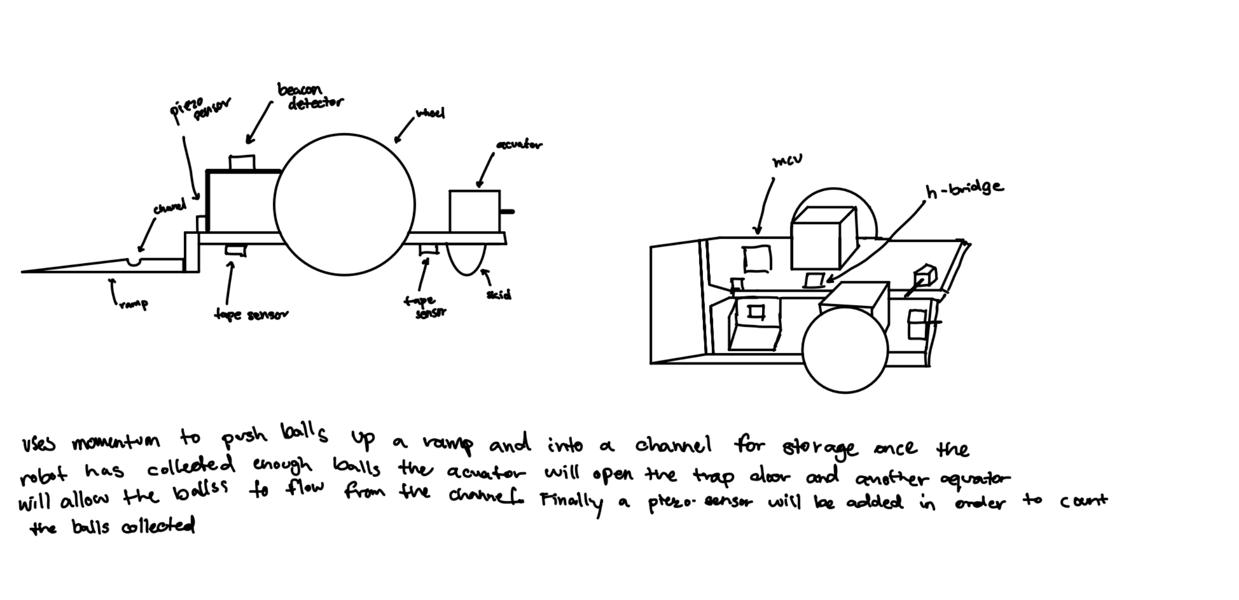


Mechanical

Initial Designs:

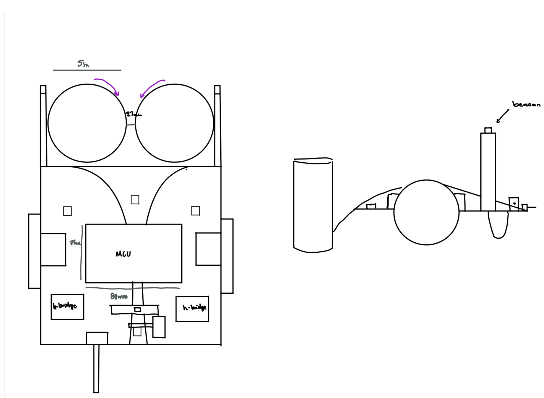
These were the initial ideas for the robot and from here we discussed the pros and cons of each design before we settled on one.

Design1: The Scooper



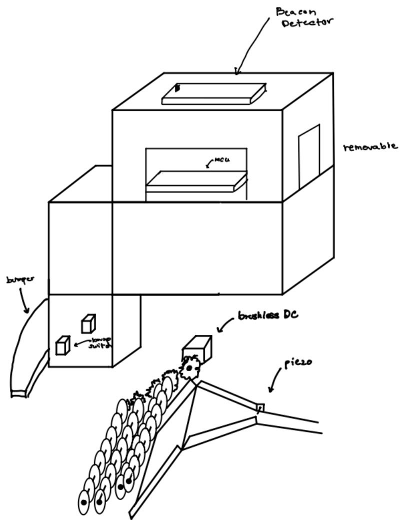
The pros of this design was that we could keep the bot low to the ground and did not need an electrical component for simply just scooping the balls. The biggest con that we were not sure where we would put the bumpers on here as well.

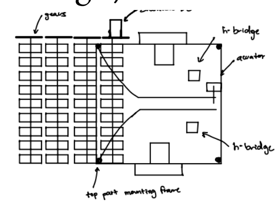
Design 2: The Roomba

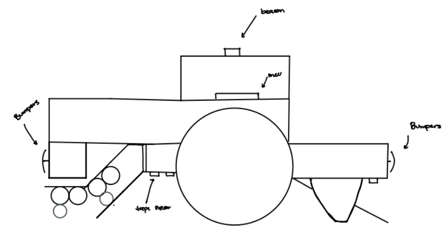


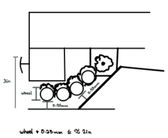
This design worked like a roomba the circles at the front of the bot represent brushes or crossed stick that would feed the balls into the bot and slide them down a channel

Design 3: The Roller(Chosen Design)





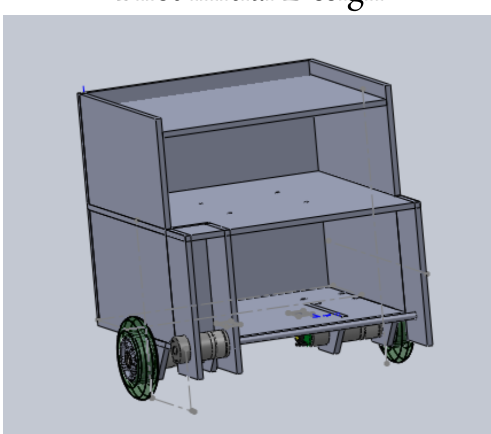




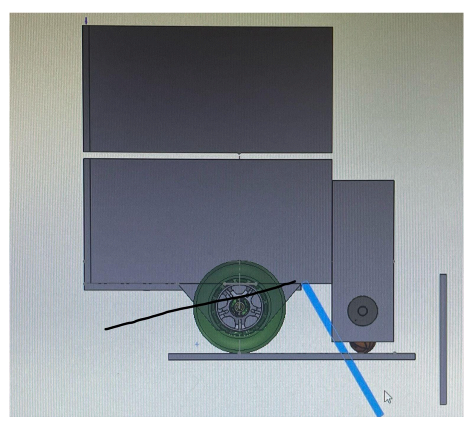
Starting Designs:

CAD of The Roller

First initial Design



This design was the barebone idea of the Roller. This rough idea allowed us to see a 3D version of the robot and with discussion made some changes. We saw that the idea of consuming the balls through the front and releasing through the back called for ramps. So we had to implement a lower base which allowed the potential energy of the balls to act as the release mechanism when the back flap is opened.

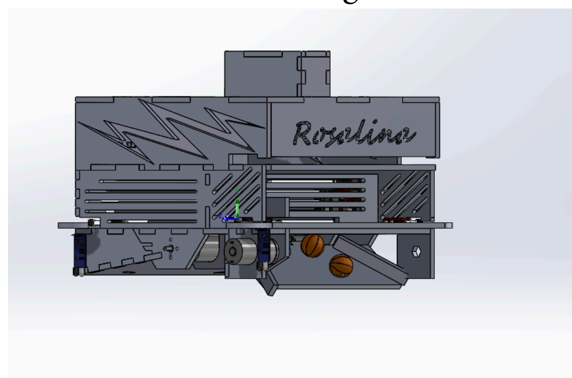


Second Design

This was the design of the first with the modifications discussed. Along with some weight reduction design.

The very bottom layer is a way for the balls to be expelled from the robot. The base above that is there to allow the motors and bumpers to be levels and easier to implement. The second layer is to hold the Uno as well as the H-bridge. The very top layer is where the beacon detector rests along with voltage regulators and perfboards that allow up to see if tape sensors or bumpers have gone off.

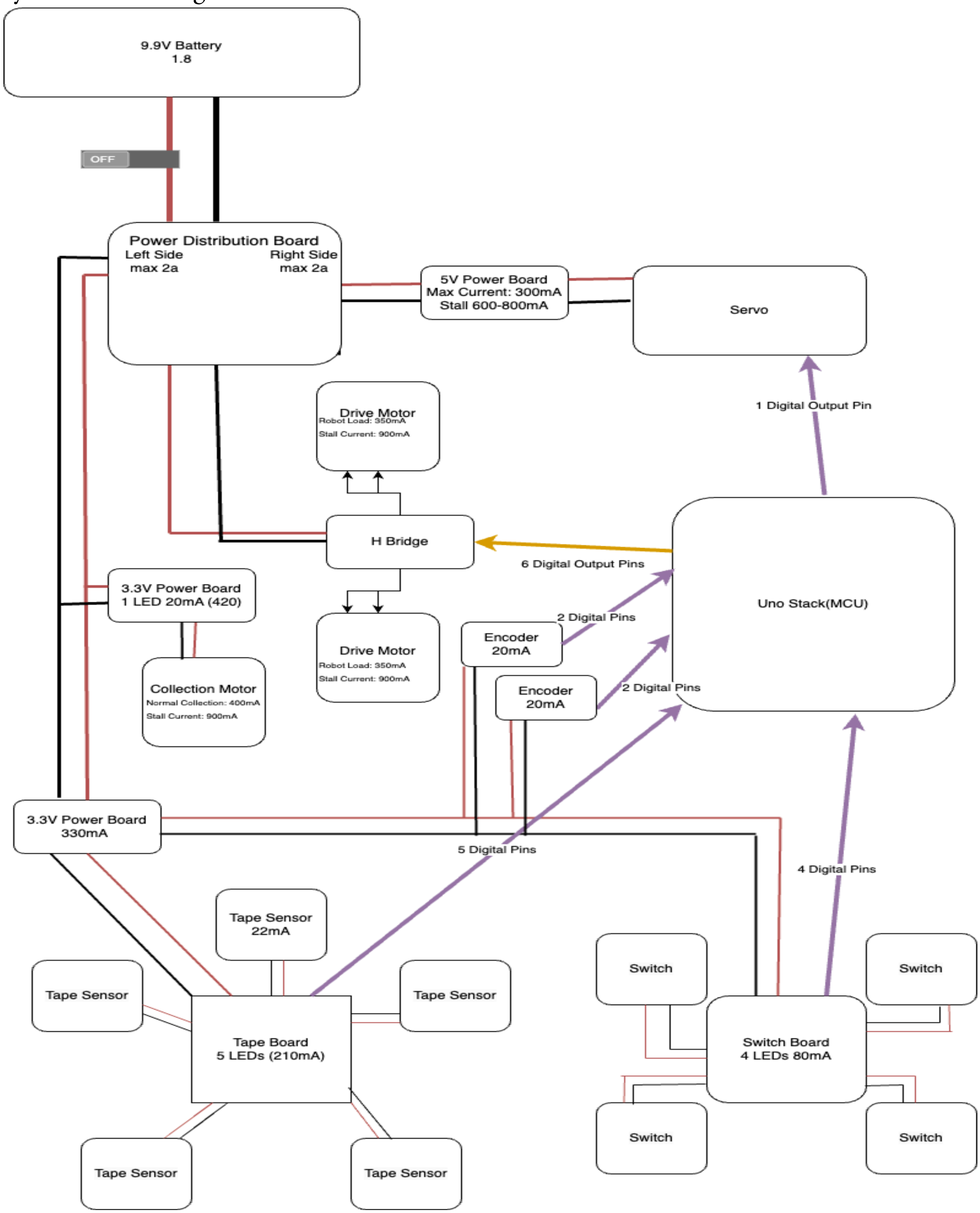
Final Design



After checking that software and electrical components were working correctly the robot underwent one more design change to fix any mechanical changes we made while testing the robot with the first two prototypes.

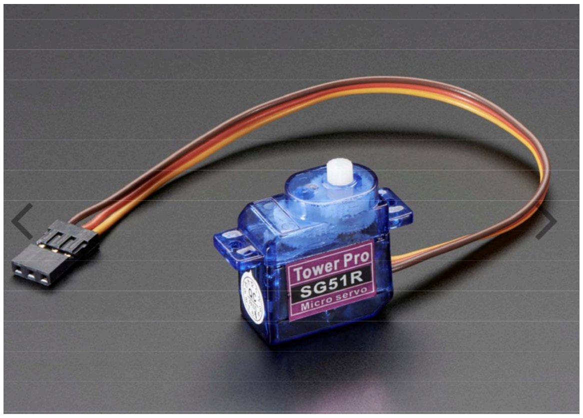
Electrical

System Block Diagram:



Motors, Sensors and Switches Used:

Output Mechanism



10

Sub-micro Servo - SG51R

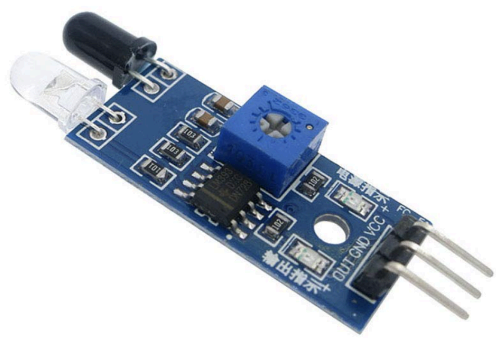
Intake Motor

4.4:1 Metal Gearmotor 25Dx48L mm HP 6V

Drive Motors

47:1 Metal Gearmotor 25Dx67L mm HP 6V with 48 CPR Encoder

Tape Sensors

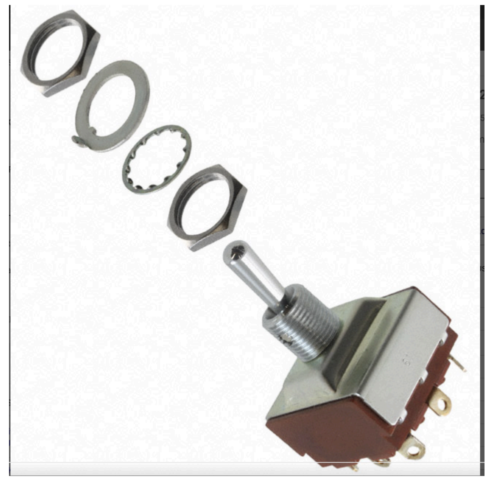


HiLetgo 10pcs IR Infrared Obstacle Avoidance Sensor

Bumper Switch

V-156-1C25 Lead Limit Switch SPDT Switch

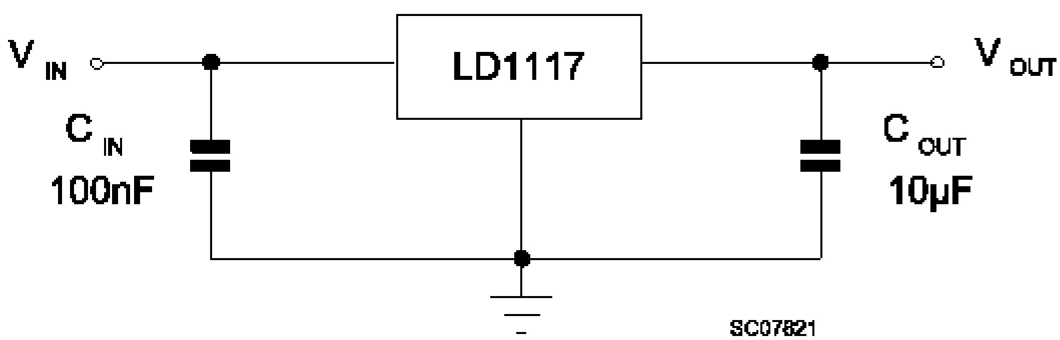
Master Power Switch

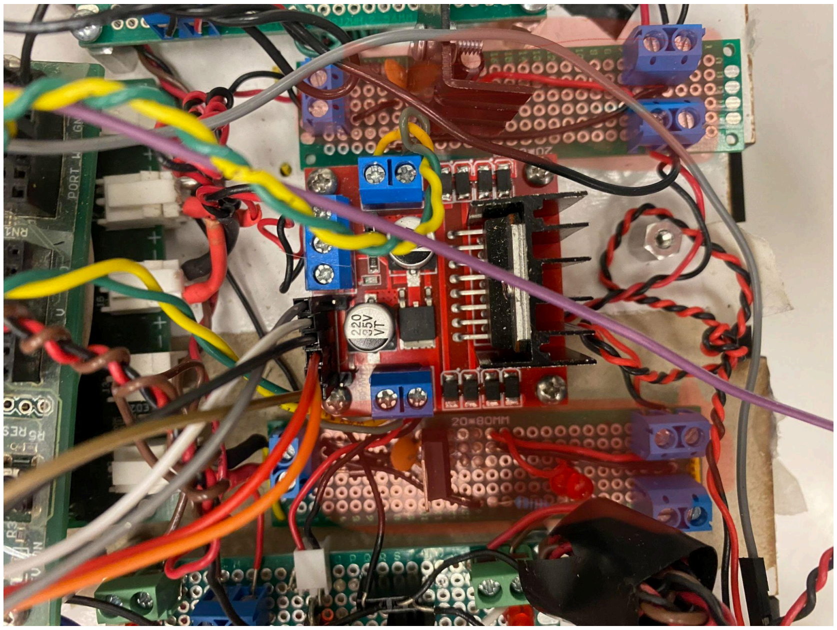


S32 NKK Switch

Regulators & Breakout Boards:

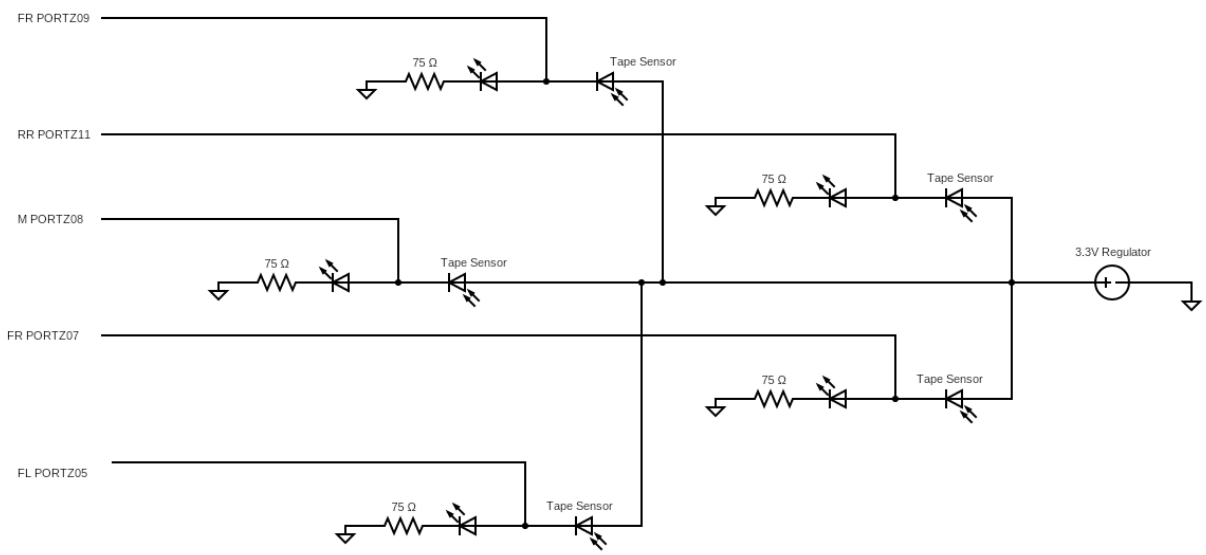
Regulators

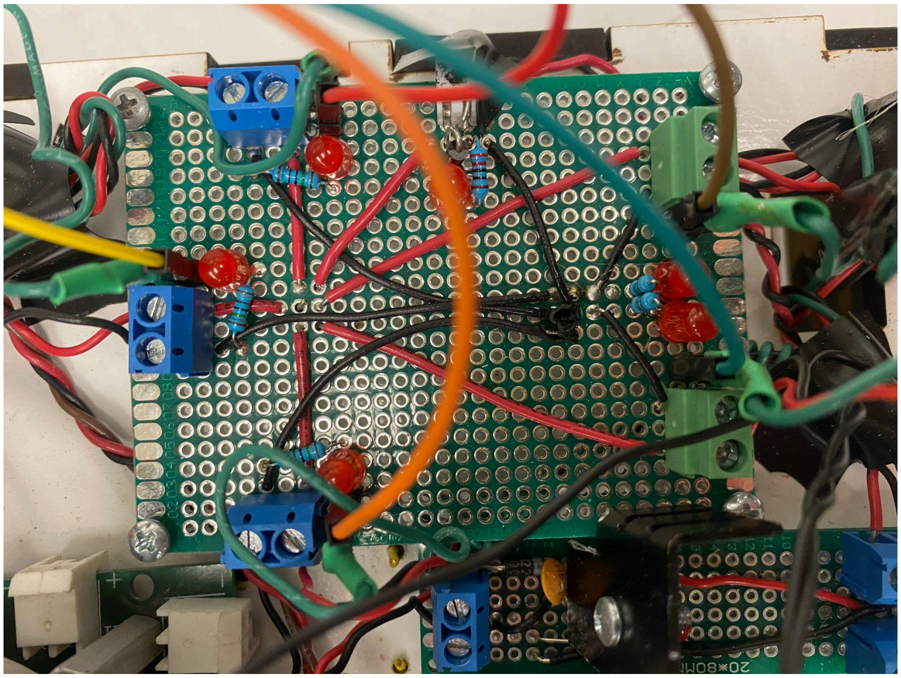




Regulators highlighted in red, include a diode for reverse polarity protection and an LED to indicate if power is being provided in order to help debug. The 3.3V regulator also has a heatsink in order to help with heat dissipation due to the large current draw of the 5 tape sensors.

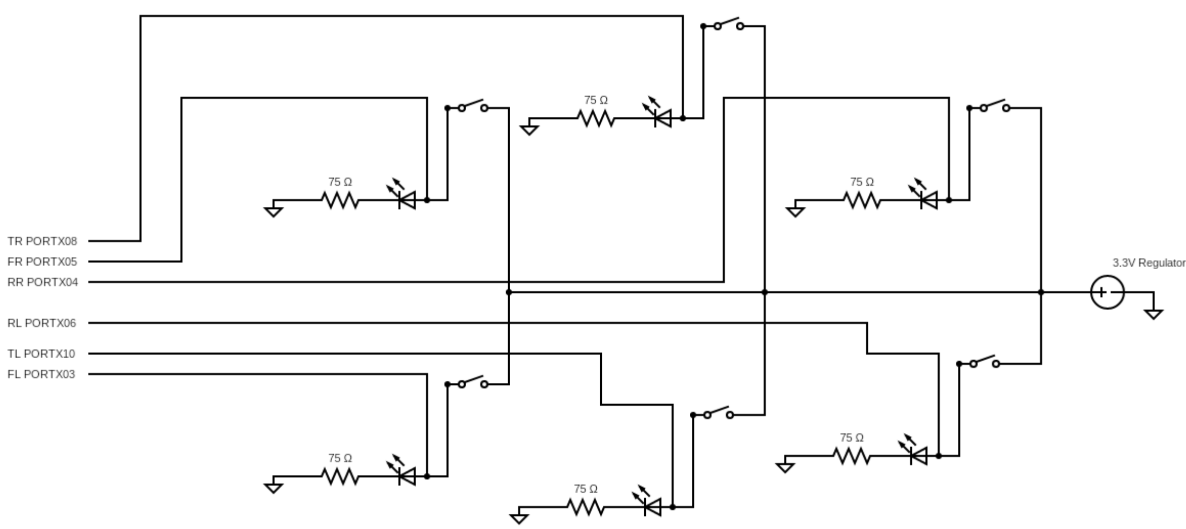
Tape Sensor BreakOut Board

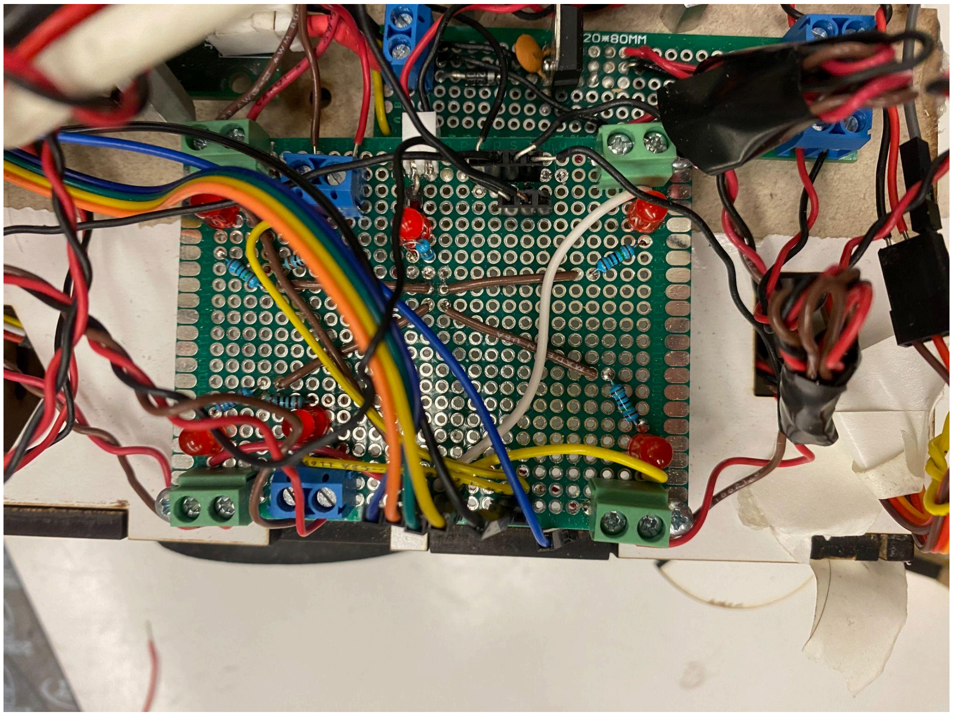




Includes a diode for reverse polarity protection and an LED to indicate if power is being provided in order to help debug.

Bump Sensor BreakOut Board





Includes a diode for reverse polarity protection and an LED to indicate if power is being provided in order to help debug.

In this project, the electrical work involved several key components and considerations to ensure the autonomous robot operated as intended.

Tape Sensor and Bumper Breakout Boards

- Tape Sensor Breakout Board: Integrated with reverse polarity protection and an LED indicator to confirm power supply, aiding in debugging.
- Bumper Breakout Board: Similarly equipped with reverse polarity protection and an LED indicator for power status, facilitating easier troubleshooting.

Voltage Regulators

- 5V Regulator: Provided a stable 5V supply for the servo which facilitated the opening of the door as well as the depositing of the balls.
- 3.3V Regulator: Used to power the sensors as well as the encoders' hall sensors.

H-Bridge for Motor Control

An H-bridge was utilized to control the drive motors, enabling forward and reverse motion as well as speed control. This was crucial for the robot's maneuverability and precise control during navigation.

Encoder and ISR for Closed Loop Movements

An encoder was initially integrated to facilitate closed-loop control of the robot's movements. An Interrupt Service Routine (ISR) was written to handle encoder signals; however, integrating this into the events and service framework proved to be too complex, leading to the abandonment of this approach.

Reverse Polarity Protection and LEDs

All circuits were designed with reverse polarity protection diodes to prevent damage from incorrect power connections. Additionally, LEDs were included in each circuit to indicate power status, significantly aiding in the debugging process and ensuring the robot operated correctly.

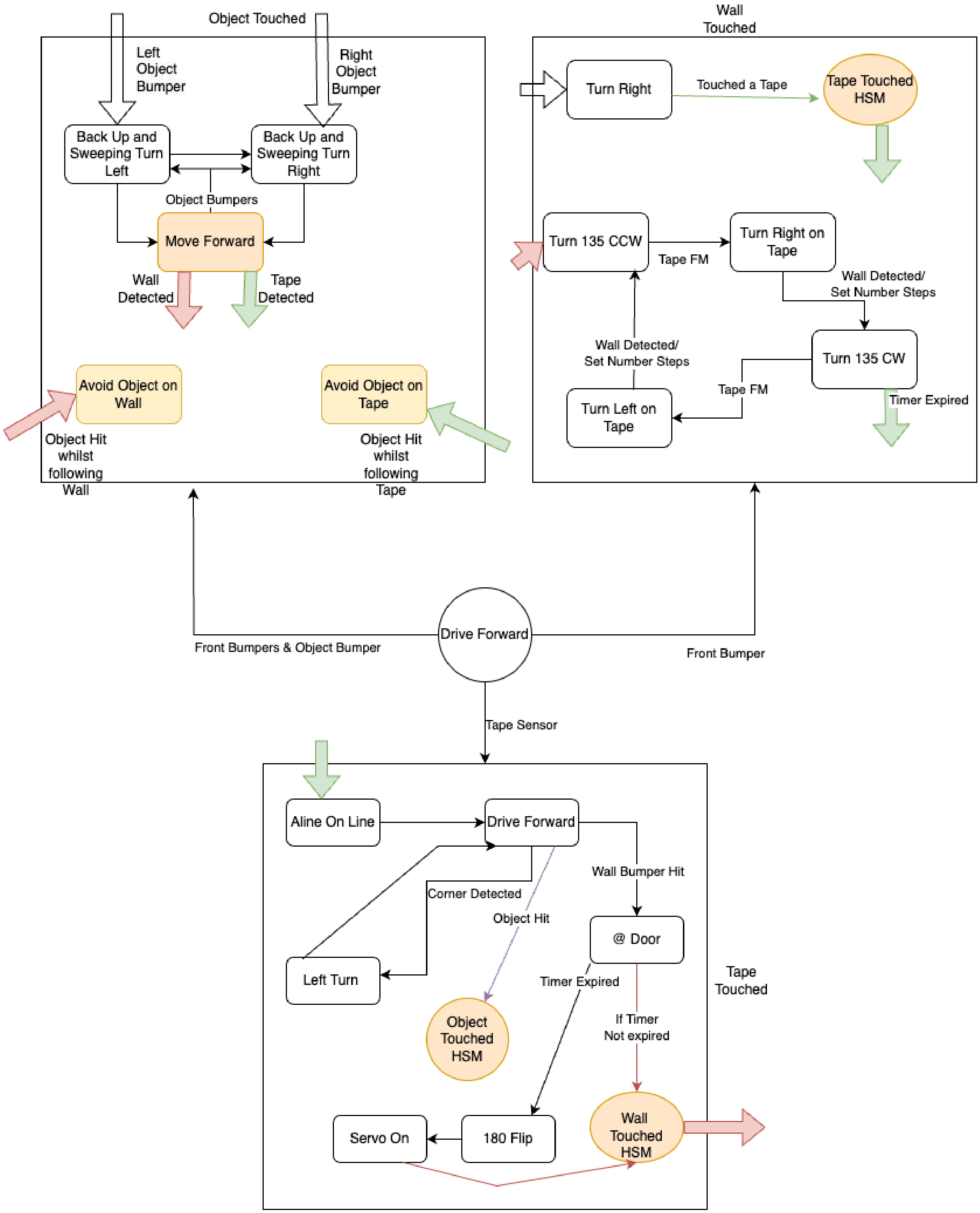
Noise Reduction for Sensor Signals

To mitigate noise interference, sensor signals were wrapped in ground wires as they connected to the microcontroller. This practice reduced the likelihood of noise affecting sensor readings, ensuring more reliable performance of the robot.

Result

These electrical design considerations and implementations were pivotal in ensuring the autonomous robot could effectively navigate the game field, detect and collect chrome balls, and perform its tasks reliably. The inclusion of protective measures and indicators facilitated easier debugging and maintenance, contributing to the overall success of the project.

Software Logic



The Hierarchical State Machine (HSM) diagram illustrates the robot's states and transitions. The main state is "Drive Forward," where the robot moves forward continuously and transitions to other states based on inputs from tape sensors and bumpers. When the left or right object bumper detects an object, the robot transitions to the "Object Detected HSM," where it backs up and performs a sweeping turn in the appropriate direction before resuming the forward state. If a wall is detected, the robot enters the "Wall Touched HSM," where it can execute a series of turns (right, 135 degrees counter-clockwise, or 135 degrees clockwise) to navigate away from the wall. Tape detection triggers the "Tape Touched HSM," where the robot adjusts its path to stay within the field, responds to corner detection, or handles ball release when at the designated door area. Additional actions include aligning on the detected tape line, making left turns, and activating the servo for ball release or orientation adjustments. The HSM's structured approach ensures the robot can effectively navigate, avoid obstacles, and handle ball collection and release tasks efficiently.

Initial Software Implementation

Basic Navigation:

In the first iteration of the robot's software implementation, the robot utilized a combination of bumper switch sensors and tape sensors to navigate its environment. Here's how it worked:

Bumper Switch Sensors:

- **Placement**: Two bumper switch sensors were mounted at the front of the robot.
- **Function**: These sensors detected physical contact with the wall or the object.
- **Operation**: When the robot moved forward and encountered a wall, one or both of the bumper switches would get pressed. This triggered the robot to stop and change direction to avoid the obstacle. This simple mechanism prevented the robot from running into walls and helped it navigate around the environment by making random turns when an obstacle was detected.

Tape Sensors:

- **Placement**: The robot had two tape sensors at the front and two at the rear.
- **Function**: These sensors detected the presence of tape on the ground, which was likely used to mark boundaries or specific areas within the field.
- **Operation**: The tape sensors detected the contrast between the tape and the floor. When the robot moved over the tape, the sensors signaled it to change direction. This prevented the robot from leaving the designated area or getting stuck in loops.

Navigation Logic:

- **Random Navigation:** The robot did not have any sophisticated localization or mapping capabilities. Instead, it relied on a basic random navigation strategy.
    - **Obstacle Avoidance:** Upon detecting a wall via the bumper switches, the robot would randomly choose a new direction to move. The rotation was usually the opposite direction from the wall.
    - **Boundary Detection:** When the tape was detected by the sensors, the robot would again choose a new direction, ensuring it stayed within the intended boundaries.

By combining the input from the bumper switch sensors and the tape sensors, the robot could perform basic random navigation around the field. This approach allowed it to avoid obstacles and stay within the designated area most of the time, even without advanced localization techniques. The simplicity of this setup made it effective for basic navigation tasks in a controlled environment.

Issues with the Initial Implementation:

The initial implementation of the robot's navigation had several notable issues:

1. Corner Detection Failure:

- **Problem**: The robot was unable to identify corners within the field created by the tape boundary.
- **Effect**: Sometimes, the robot would not turn enough when approaching a corner, causing it to get stuck in a loop of trying to change directions.
- **Consequence**: This resulted in the robot eventually falling off the field because it could not correct its course effectively in corners.

2. Looping During Obstacle Avoidance:

- **Problem**: The robot could get stuck in a loop when trying to avoid obstacles.
- **Effect**: Since the navigation was based on random direction changes without a planned path, the robot could repeatedly turn in a manner that brought it back to the same obstacle or area.
- **Consequence**: This looping behavior meant that the robot could waste time, making it inefficient and potentially getting stuck indefinitely in some scenarios.

3. Lack of Ball Release Logic:

- **Problem**: There was no logic implemented for ball release.
- **Effect**: While the robot could intake balls, it had no logic to determine when and where to release them.

- **Consequence**: This limitation meant that the robot could collect balls but not complete its intended task of placing or releasing them at the designated location, reducing the overall effectiveness and functionality of the system.

Final Software Implementation

For the final iteration of the robot, several enhancements were made to address the issues from the initial implementation. These improvements included better sensor placement, a more structured navigation path, enhanced obstacle avoidance, the addition of ball release logic, and the use of a Hierarchical State Machine (HSM) to manage the robot's states. Here's a detailed explanation:

Bumper Switch Sensors:

- **Lower Sensors:** Two bumper switch sensors were mounted at the front and lower position to detect walls. These sensors were effective at identifying low-lying obstacles like walls, enabling the robot to stop and change direction before colliding.

- **Higher Sensors:** Two additional bumper switch sensors were mounted higher above the lower ones to detect objects that might be positioned above the ground level but within the robot's path, such as tables or chairs. This ensured the robot could detect and avoid a broader range of obstacles.

- **Rear Sensors:** Two Bumper switch sensors were mounted to the rear of the robot to detect any obstacle while the robot was reversing.

Tape Sensors:

- **Front Sensors:** Three tape sensors were installed at the front: one on the front right (FR), one on the front left (FL), and one toward the front center. The FR and FL sensors helped detect the edges of the field, while the center sensor was crucial for detecting corners. This configuration allowed the robot to navigate the boundaries of the field accurately and avoid going off course.

- **Rear Sensors:** Two tape sensors were mounted at the rear to detect the tape boundary at the back of the robot. These sensors ensured that if the robot moved backward, it could still recognize when it was approaching the edge of the field.

Navigation and Obstacle Avoidance

- **Differentiation Capabilities:** The combination of higher and lower bumper switches, along with strategically placed tape sensors, allowed the robot to differentiate between walls, objects, and the tape edges and corners effectively. This ensured the robot could accurately identify and respond to different types of obstacles and boundaries, improving its overall navigation capabilities.

- **Navigation Path**: The robot followed a counter-clockwise navigation path around the field. This predefined path made the robot's movements consistent and predictable, reducing the chances of getting stuck or wandering aimlessly. The structured path also helped in systematically covering the entire field, ensuring efficient navigation.

- **Obstacle Avoidance:** Inspired by the Vector Field Histogram (VFH) object avoidance algorithm, the robot performed simple object avoidance by calculating a new path around the obstacle based on continuous sensor input. Once the obstacle was avoided, the robot would merge back into its initial navigation path. This ensured that the robot could navigate around obstacles without losing its way or deviating from its path.

Ball Handling Logic

- **Ball Release:** Ball release logic was implemented to allow the robot to release collected balls after every two passes around the trap door opening. This meant that after collecting balls during its navigation, the robot would deposit them at designated intervals, ensuring efficient ball handling and meeting the operational requirements.

- **Optimization for Ball Collection:** To meet the 30-ball collection requirement, any balls collected after the two designated releases were held within the robot. This approach optimized the robot's performance by preventing unnecessary ball release attempts, thus focusing on achieving the primary goal of collecting the required number of balls.

Issues with the Final Implementation:

1. Corner Detection Redundancy:

Problem: There was no redundancy for detected tape corners. The robot relied on a single corner tape sensor to detect the corners of the field.
Effect: If the corner tape sensor failed to activate due to varying light conditions or sensor malfunction, the robot had a high possibility of going off the field. This vulnerability could lead to navigation errors, especially in less-than-ideal conditions.

Consequence: This lack of redundancy made the robot's corner detection less reliable, potentially causing it to fall off the field and fail to navigate properly. Addressing this issue would require implementing additional sensors or improving the existing ones to ensure consistent performance.

Competition Video

Conclusion

Overall, this project proved difficult with the integration of electrical, mechanical, and software components needed to create a functional autonomous robot. Completing the ECE 118/218 Mechatronics Final Project required dedicated teamwork, careful planning, and continuous problem-solving.Each week, we met the check-off milestones to keep our project The design reviews provided useful feedback, helping us refine our prototypes and address potential issues. The final competition and public demonstration showcased our robot's capabilities. This event tested our technical achievements and our ability to explain our work clearly. In conclusion, this project taught us the importance of thorough documentation, effective time management, and teamwork. It provided us with practical skills and a solid understanding of the mechatronics field, preparing us for future engineering challenges.