Final Documentation & Schematics: Grape Expectations 🍇

Senri Nakamura, Naomi Arnold, Dani Price, Alex Moy

Objective

In this project, we were tasked with designing and building a small robotic vehicle capable of autonomously navigating a test track by following colored lanes. The system integrates multiple components, including an Arduino, color sensing, motor control, obstacle detection, and battery monitoring circuits. The course emphasizes the full engineering design process, from ideation and iterative testing to refinement. Successful completion involves collaboration, project management, and effective communication with the client. Ultimately, this project prepares us to think critically, solve complex problems, and work as design engineers.

Pictures of Bot

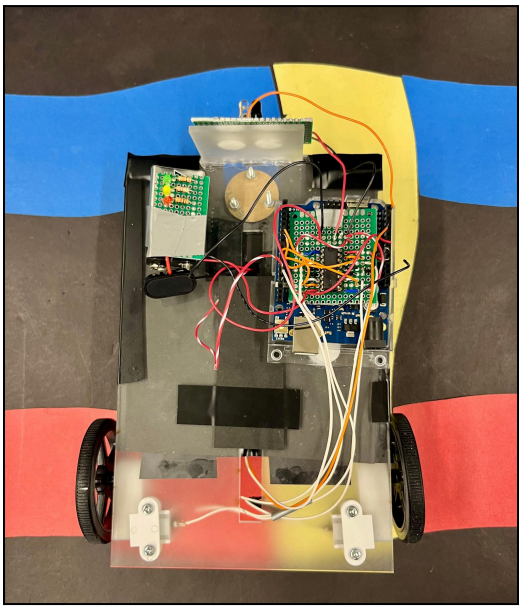

Figure 1: Overhead view

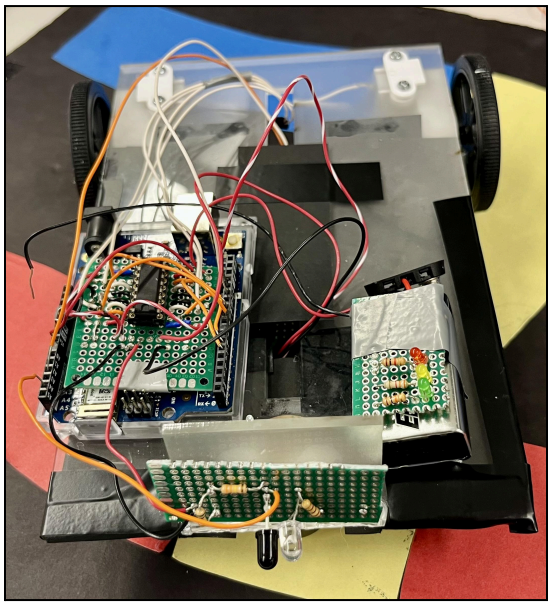

Figure 2: Front view

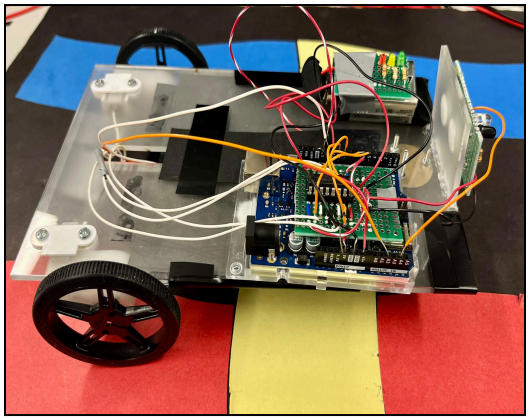

Figure 3: Side view

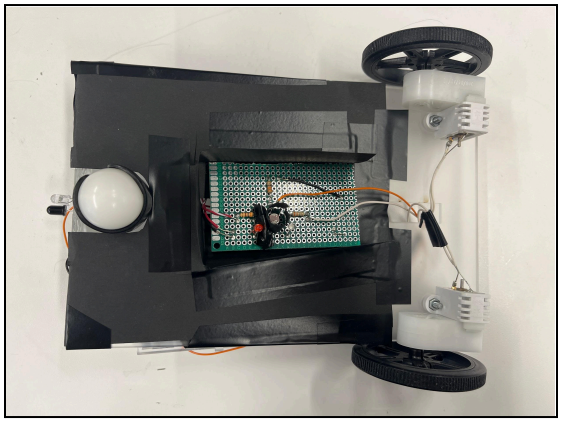

Figure 4: Bottom view

Bot Specifications

The parts that we used to build the bot consist of multiple circuits, an Arduino, and the physical chassis that holds all the parts together.

Motor and H-Bridge

To move our bot, we use an L293D bridge integrated circuit (IC), which routes current flow to start, stop, and reverse the motor. The circuit had multiple pins that, when powered, did different actions. There was a logic voltage, and a motor voltage, with enable pins, and pins that were for the motor. After setting up the circuit on the breadboard, we had to connect it to the Arduino, which took in input pins, three from each side for the left and right motors. Below is the block diagram of how the H-Bridge would connect with the motors.

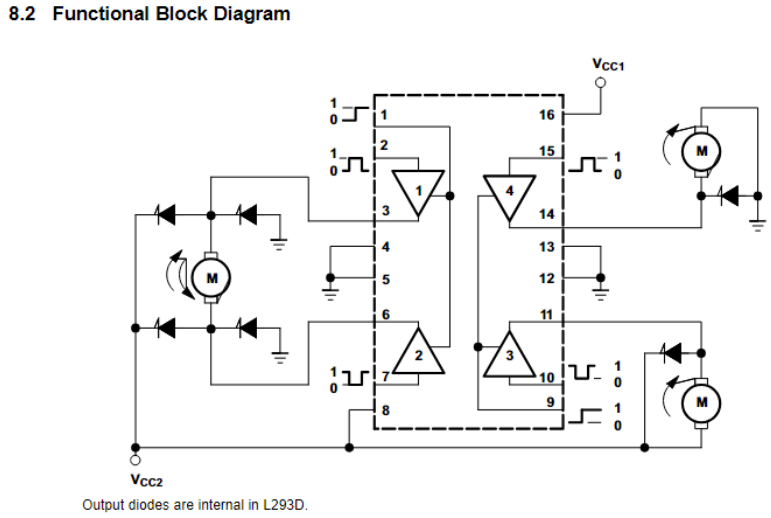

Figure 5 H-Bridge Block Diagram

Color Sensing Circuit

The color sensing circuit uses a CSS001-8001 Jameco Photocell in combination with a blue LED and a red LED to determine the color of the surface. The circuit works by toggling the blue and red LEDs and measuring the reflected light off the surface with the photoresistor. The different colors reflect varying amounts of light due to their wavelengths, leading to a voltage difference from the photoresistor. This voltage difference is read in by the Arduino. Below is a

circuit diagram of our color-sensing circuit. The output of the circuit is fed into pin A2 of the Arduino, allowing it to process the analog signal. Pin 13 on the Arduino controls the blue LED, while pin 12 controls the red LED, toggling them on and off for color detection.

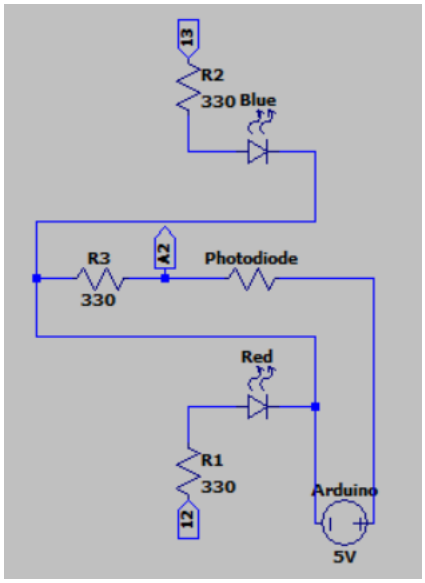

Figure 6 Color Sensing Circuit Design

Obstacle Detection Circuit

The obstacle detection circuit uses an LVIR3333 IR LED and an SFH 203 FA photodiode 900 nm to detect nearby objects by measuring reflected infrared light. The infrared LED emits light, and the photoresistor captures the reflected light when it encounters an obstacle. In order to gain a higher voltage output, multiple resistors were used. The output of the circuit is fed into pin A4 of the Arduino, allowing it to process the analog signal for obstacle detection.

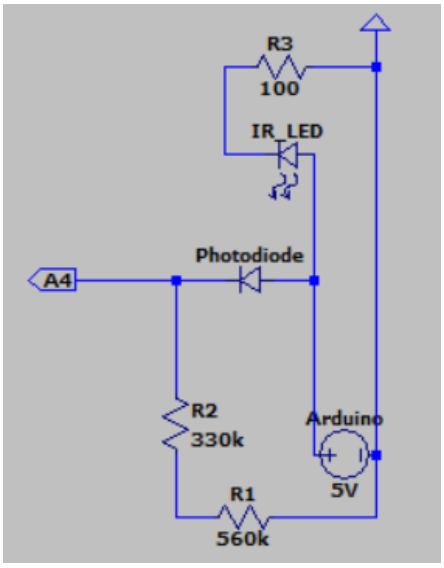

Figure 7 Obstacle Sensing Circuit Design

Battery Detection Circuit

The battery detection circuit uses voltage dividers to monitor the bot's battery level and control an LED indicator system. Depending on the battery voltage, the LEDs would light up, where at full charge, all LEDs are lit, indicating optimal battery status. As the voltage drops, the LEDs turn off one by one from green to yellow to red. When only the red LED is on, it would indicate that the battery requires a replacement.

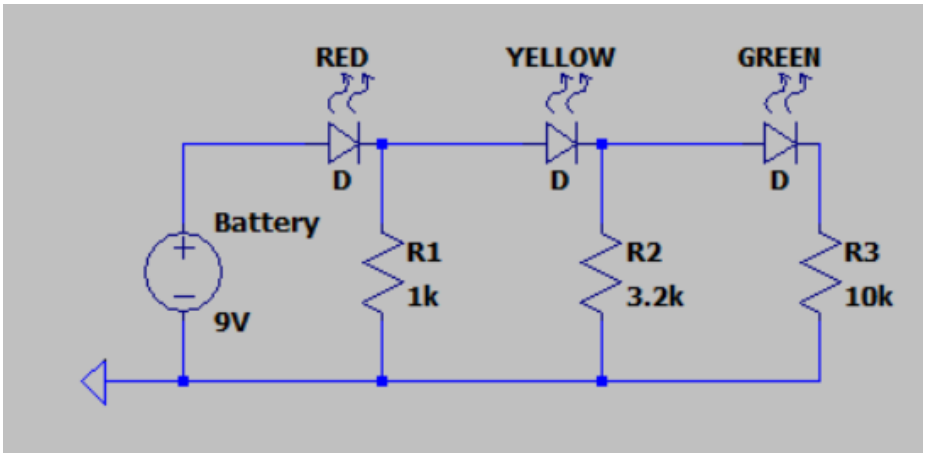

Figure 8 Battery Detection Circuit Design

Chassis

The bot's chassis was designed using the Onshape tool, and laser cut at Nolop. The base material was made from acrylic and contained pre-designed holes made for screws to mount the motors and the Omniball. To mount the circuits, we used Velcro, which allowed for more flexibility in circuit placement and accessibility in case of a malfunction to any circuit component. We had to go through multiple designs to end up with the final product since there were multiple components to consider. Once the final circuit designs were completed, it became easier to identify how each component would interact and thus where it should be located on the chassis. Through this process, we managed to create a final product that was both functional and well-organized.

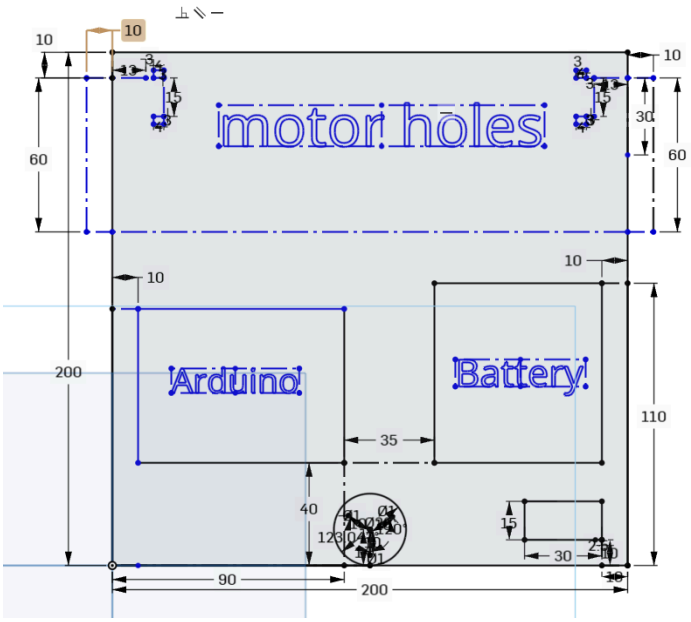

Figure 9 Chassis Design

Total Parts Used

Motor Control	**Battery Detection**	**Color Sensing**	**Obstacle Detection**	**Physical Bot**
L293D bridge integrated circuit (IC)	Red LED Green LED Yellow LED Resistance 10k 3.3k 1k	CSS001-8001 Jameco Photocell Red LED Blue LED Resistance 300 330 330	Resistance 560k 330k 100 LVIR3333 IR LED SFH 203 FA photodiode 900nm	Motors Pololu Ball Caster w/ 1” plastic ball Pololu Wheel 40x7mm Pololu Mini Plastic Gearmotor Bracket Acrylic Chassis Arduino 9-volt battery

Arduino Schematics

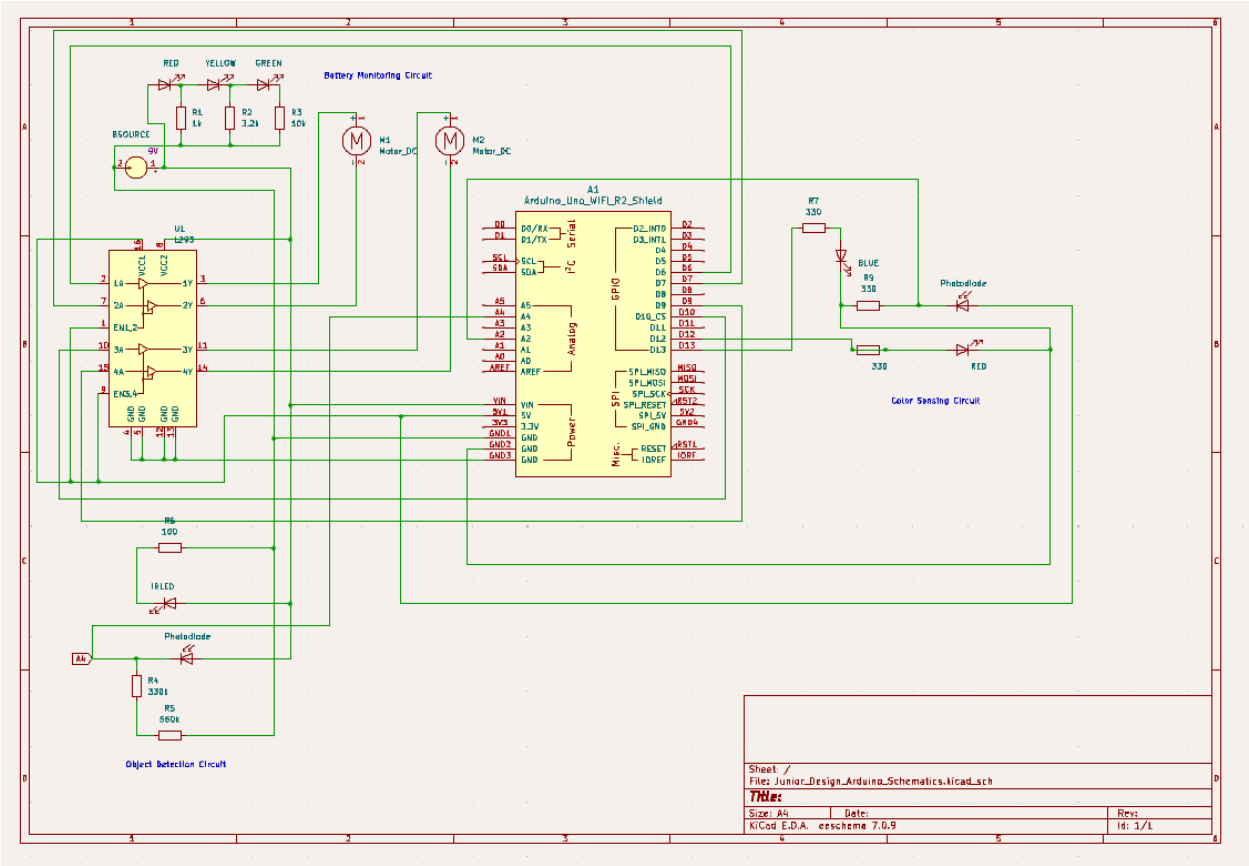

Figure 10 Arduino Schematics with Battery Detection, Obstacle Detection, and Color Sensing Circuits

Customer Response

During the bot design process, we consistently thought of how to deliver the optimal product to our customers. During and after all the components were combined, we considered the customers' desires in how one would view the bot and utilize its capabilities. We believe that one of the customers' primary concerns was ensuring the bot's battery level could be monitored effectively to prevent interruptions in operation. To address this problem, we implemented an intuitive LED indicator system. The LEDs would change color to indicate the battery level of the bot. All the LEDs lit indicate the battery is at an optimal level. When the green LED turns off, the battery is in operating range, but caution is advised to the customer. When the yellow LED turns off (only the red LED is on), this means that the customer would need to replace the battery. This visual cue produces an intuitive understanding of the bot's battery status.

Customers emphasized the importance of a clean and professional appearance for the bot. To meet this need, we focused on minimizing the amount of exposed wiring by creating a neat design. In attempting to achieve a cleaner aesthetic, we made a hole in the middle of the chassis to clean up the wiring. This approach allowed us to internally route the wires, maintaining an organized appearance without visible clutter. As a result, it would prevent wires from hanging over the edge, or become excessively long, which would adversely affect the bot's appearance.

Additionally, security is a major concern, particularly regarding the bot's ability to receive and act on messages sent through a server. To mitigate potential risks, we chose to attack this issue by only accepting trusted server IDs. If the server received messages that were not from recognized IDs, our server communication code blocked the message. Otherwise, the messages from trusted IDs would be able to go through and give the bot instructions. This approach makes sure that unwanted commands or hacking attempts are mitigated, ensuring the integrity and reliability of the bot.

Code

GitHub Link: https://github.com/naomi-arnold/grape_expectations
Server ID: 89C87865077A
(Use the server ID to log into the remote and or the WebSocket server)

User Instructions

To replicate our success in navigating the test track, the customer simply needs to log into the WebSocket server and enter either "Bot 1" or "Bot 2" into the message field. Once they make their selection, the system will automatically configure everything required, including track navigation settings and the sensors required. After the setup is complete, the chosen bot path will begin navigating the track based on the predefined track configuration, enabling the user to replicate the navigation successfully.