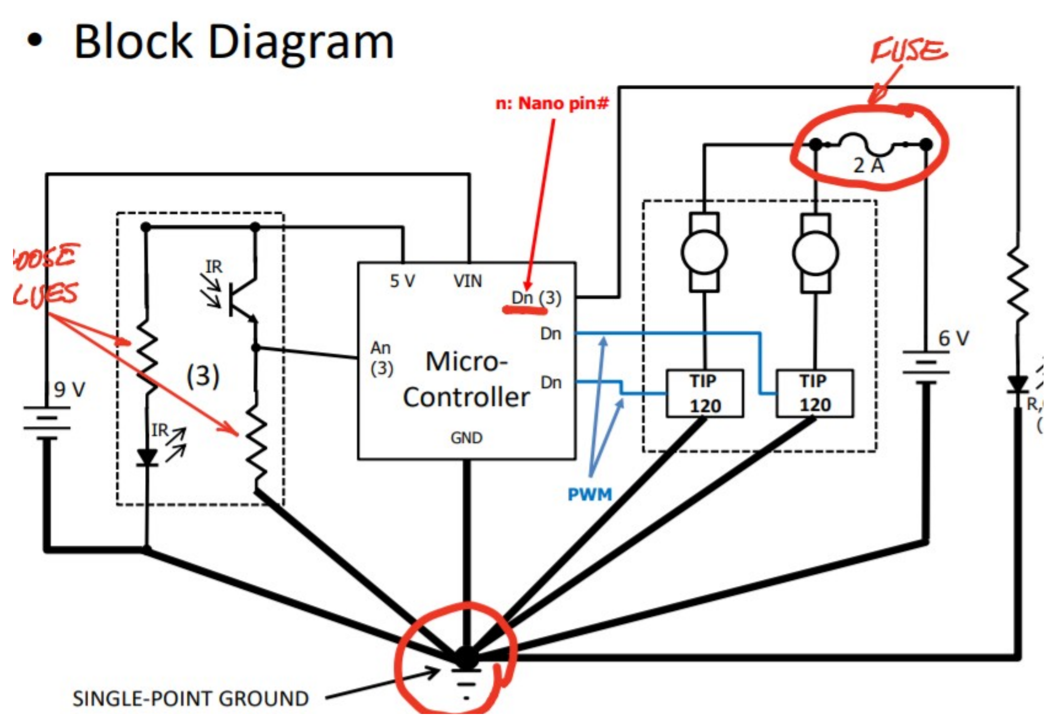
EE3 Final Project Report
Khyle Calpe, Calvin Kuo

## Introduction

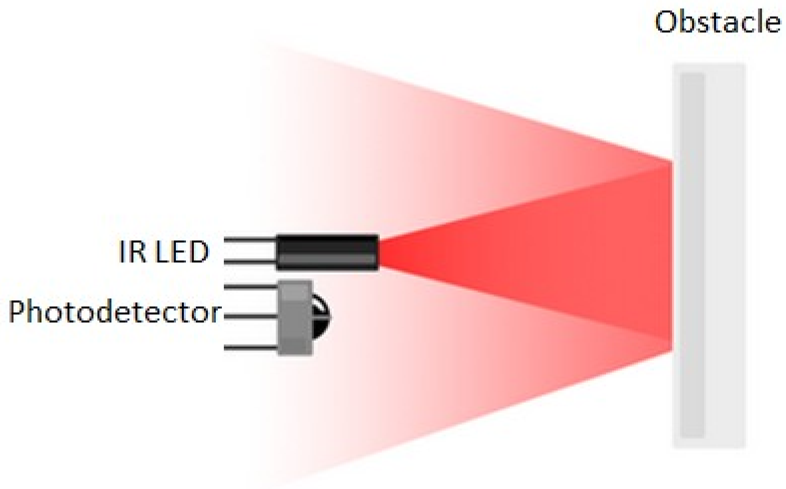
The overarching goal of the project was the design of a small electric car that autonomously navigates through a path indicated by a black line on top of a white background. To reach this goal, my group divided the work necessary into smaller goals. In chronological order, the car needed the ability to control the speed of two calibrated motors, detect and move along a path, and light up LEDs to indicate the turns.

The car uses an Arduino Nano microcontroller to alter the speed of two motors through two TIP120 transistors and pulse width modulation (PWD). The speeds of both motors needed to be measured at a certain duty cycle. Then, the PWD values are manually adjusted through the Arduino IDE based on the speed values. Next, the car uses three infrared emitters and three infrared (IR) sensors to detect a path in three different directions (left, middle, and right). Based on the sensor's input from the environment, measurements were taken to ensure that the Arduino Nano alters the speed of the motors with respect to the direction of the path. For instance, a path detected in the left will cause the left motor to retain its speed while speeding up the right motor to turn left. Accordingly, three LEDs (left, green, red) lit up to indicate the turns of the car. For instance, a left turn lit up the blue LED.




*The block diagram shows the schematic of the circuit used for the entire car. On the right side, the diagram shows the connection between the Arduino Nano, two motors, two TIP120 transistors, a 6 V battery, three LEDs, and a fuse. On the left side, the diagram shows the connection between the Arduino Nano, a 9 V battery, one IR emitter, and one IR sensor. The actual car contains three emitters and three sensors. [1]

Testing Methodology (Path Sensing Subsystem)



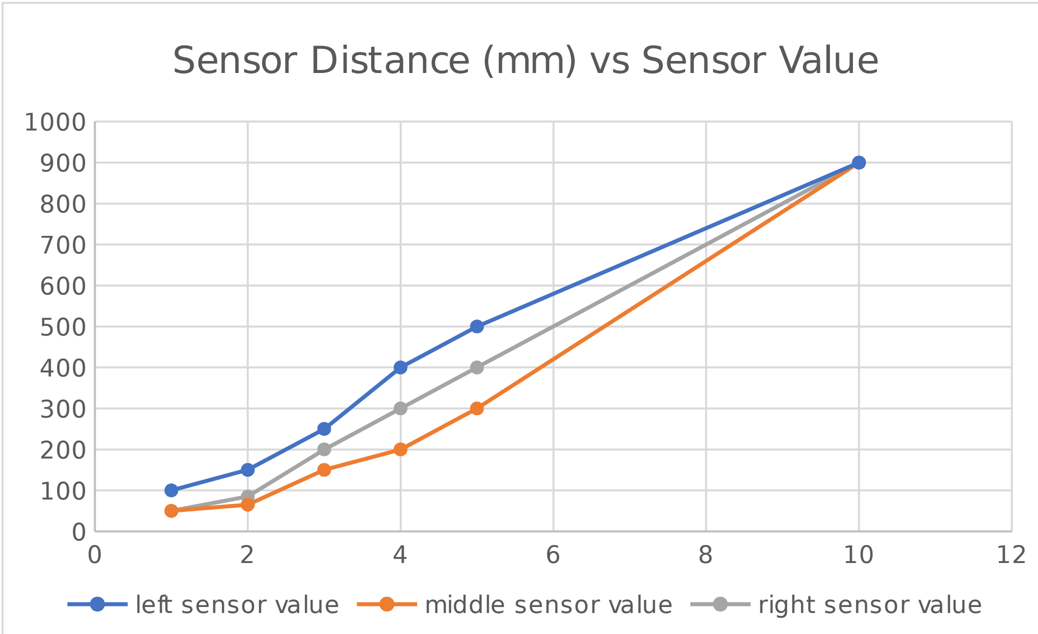

*Visual representation of the test. The intensity of the rays reflected from the obstacle (path) will vary according to the color of the obstacle (path) and the distance between the IR components and the obstacle (path). Taken from python-exemplary.com.

How We Designed the Test

To ensure that the car properly senses the path, my group focused on testing the functionality of the IR emitters, sensors, and the circuitry between the IR components and the Arduino Nano. For the design of the test, my group set up the Arduino IDE to print out values based on the three sensors' inputs on the IDE's serial monitor. Then, my group placed the sensors on top of a surface colored black on one side and white on the other side. If successful, the Arduino IDE's serial monitor will display a range of values that indicated a clear contrast when the sensors are detecting black or white. My group recorded the values for each sensor and set up the Arduino Nano to slow down or speed up a certain motor based on these values.

How We Conducted the Test

After setting up the car with all the components mounted on the car, my group implemented the test for IR sensing with a surface that contained black and white colors. My group uploaded the test program in the Arduino Nano IDE and checked for the values of each sensor. Upon setting both black and white colors under the car's IR sensors, my group tested for input values under varying distances. My group decided on maximizing the sensitivity for each sensor by checking for the maximum range of values that showed that the sensor detected a black color. My group only did this once for each sensor since only a change in wiring can cause an error to occur.




*The figure shows the values recorded for the distance between each sensor and the input value. The cutoff occurs at 5 mm since that is the maximum distance between the IR sensor and the track. After 6 mm, the values are treated as if the sensor is detecting a white color. Hence, the car will not respond to values that correspond to 6 mm and more.

<u>How We Analyzed the Test Data</u>

Upon measuring the sensor values from the infrared receivers from the Arduino IDE's serial monitor, my group compared the values on a distance vs intensity line graph. From 1 mm to 5 mm (the maximum height for the IR sensor), my group mapped data points showing the relationship between distance and intensity that was usable for the project. For distances more than the car's height, my group measured the intensity of an arbitrary distance by lifting the car. My group tried to determine a sudden increase in intensity for an arbitrary distance, but none could be found except for an approximately linear growth in distance vs intensity.

<u>How We Interpreted the Data</u>

My group initially believed that the IR sensor interpreted the inputs from the IR emitter as binary values instead of a range of values. Because of the data, my group was able to assign the proper values for the Arduino Nano to interpret the input values from the IR sensor and control the car. Based on the maximum distance allowed by the car's, my group decided on the distance between the IR components and the path. Accordingly, 5 mm allowed for a sensible cutoff point. My group determined the base values for path detection on the intensity values that corresponded to the 5 mm cutoff point. From the left sensor to the right sensor, my group used the values 500, 400, and 300.

## Results and Discussion

### Test Discussion

The implementation of the IR sensors' values worked for the car during testing. Quantitatively, the values assigned to each sensor were proper since none of the IR sensors showed errors in detecting a path. For instance, if an error did occur, the car would not make a turn at a specific point. Perhaps, the IR sensor relayed higher-than-expected values due to the color fading from the path or a curvature at some point which showed the line as partially white and partially black. Fortunately, the testing showed that car, although slow, properly navigated through the path. Hence, the path sensing for the car worked as well.

### Race Day Results

During Race Day, we tested the car only once in the track and the car successfully navigated through the track. Accordingly, the car's result during Race Day corresponded with the car's results during testing. Unfortunately, the results also meant that the car navigated through the path very slowly and erratically. Furthermore, only the blue LED lit as the car navigated through the path. This problem occurred due to the constant leaning of the car towards the right side, which meant that the IR sensor on the right kept returning values that indicated left turns, which corresponded to the blue LED.

For improvements, my group attempted to both solve the blue LED issue and speed up the car. Unfortunately, both attempts did not work. The blue LED would always stay on regardless of the alterations in the code. Perhaps, the structure of the code itself presented problems on a lower level than what was observed. Additionally, attempts at speeding up the car's navigation speed always led to the car driving past the boundaries of the path.

## Conclusion and Future Works

The final designed of the car functioned properly in the sense that it navigated through the path successfully albeit slowly. However, the car leaned on one side too much, which led to problems. Despite the problems, my group learned about the functionality of numerous electrical components and the proper circuitry that connects these components together, which create things with greater functionality that apply to the real world. If given an extension, I would implement a better navigation system that speeds up the motors' speeds based on how long the car has been moving in a certain direction.

## References


1. Circuitry of the car's components from Professor Briggs's lecture from week 5.