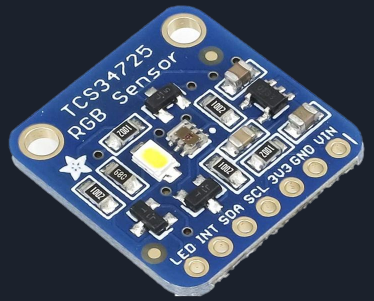
Line Following Robot - Part 2

Alfred Palacios
Daniel Delgado
Alberto Luna

Five Functions

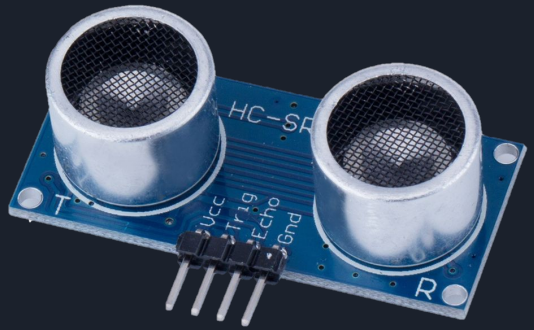
QRE113 Line Sensor

Senses black line on the floor to follow



TCS34725 Color Sensor

Senses black line to know when to stop



Ultrasonic Sensor

Senses distance so it will not crash into anything in front

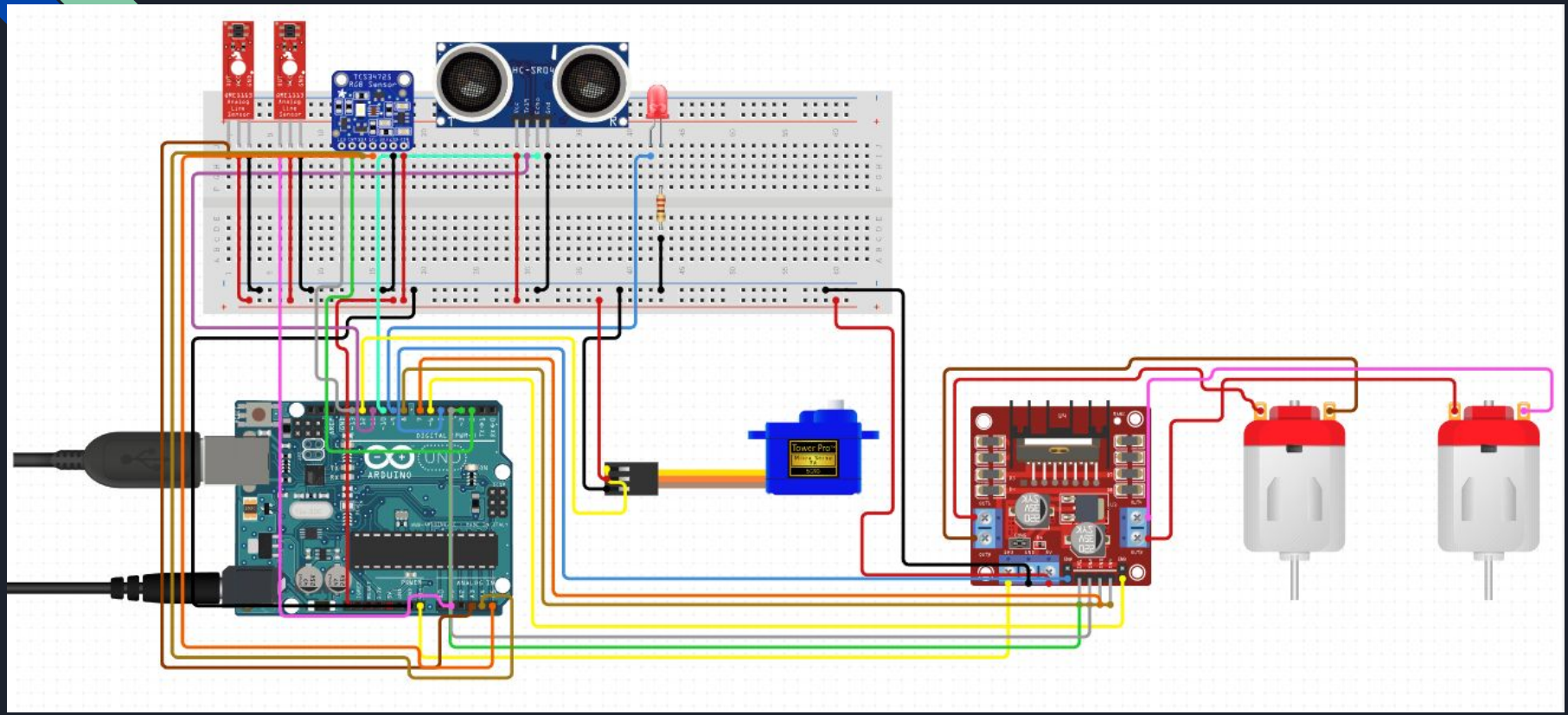
RGB LED

LIghts up different colors when robot car is in certain situations

Servo

Rotates flag pole to simulate a moving flag

Schematics



Code

- Defined libraries
- Defined variables

```
//Code for the QRE1113 Analog board
//Outputs via the serial terminal - Lower numbers mean more reflected light
#include <Wire.h> // Control I2C bus
#include "Adafruit_TCS34725.h" //Control color sensor

//Servo library
#include <Servo.h>
Servo myservo; //create servo object to control a servo

//Ultrasonic semsor library
#include <NewPing.h>
#define TRIGGER_PIN 5 // Arduino pin tied to trigger pin on the ultrasonic sensor.
#define ECHO_PIN 4 // Arduino pin tied to echo pin on the ultrasonic sensor.
#define MAX_DISTANCE 300 // Maximum distance we want to ping for (in centimeters). Maximum
// sensor distance is rated at 400-1000cm.
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // NewPing setup of pins and maximum
// distance.

/* Initialize with default values (int time = 2.4ms, gain = 1x) */
Adafruit_TCS34725 tcs = Adafruit_TCS34725();
#define CW 0
#define CCW 1
// Motor definitions:
#define MOTOR_A 0
#define MOTOR_B 1
// Pin Assignments //
// Don't change these! These pins are statically defined by Arduino & Ardumoto shield layout
const byte PWMA = 3; // PWM control (speed) for motor A
const byte PWMB = 11; // PWM control (speed) for motor B
const byte DIRA = 12; // Direction control for motor A
const byte DIRB = 13; // Direction control for motor B

int qreLeft = A0; //connected to Analog 2
int qreRight = A2; //connected to Analog 0

byte spd = 150; // forward speed
byte hiSpd = 150; //turning speed

//RGB led digital pins
int red_light_pin= 8;
int green_light_pin = 9;
int blue_light_pin = 10;

int threshold = 855;
// threshold for line sensor values. Greater than this means on the line (dark)
// less than this means the sensor is off the line (light)
// This must be determined experimentally for each surface and track
```

Code

- Motor A and Motor B setup as void function.

- Setup which initializes the servo pin, the RGB led pins, and the motor pins.

```
void setupArdumoto() {
// All pins should be set up as outputs:
pinMode(PWMA, OUTPUT);
pinMode(PWMB, OUTPUT);
pinMode(DIRA, OUTPUT);
pinMode(DIRB, OUTPUT);
// Initialize all pins as low:
digitalWrite(PWMA, LOW);
digitalWrite(PWMB, LOW);
digitalWrite(DIRA, LOW);
digitalWrite(DIRB, LOW);
}
```

```
void setup()
{
myservo.attach(6);//attachs the servo on pin 9 to servo object
myservo.write(0);//back to 0 degrees
delay(1000);//wait for 1 seconds

//RGB led pin outs
pinMode(red_light_pin, OUTPUT);
pinMode(green_light_pin, OUTPUT);
pinMode(blue_light_pin, OUTPUT);

setupArdumoto(); // Set up & initialize all motor drive pins
if (tcs.begin())
{
}
else
{
allStop();
while (1);
}
}
```

Looped Code

- Initialized variables and servo rotation.
- Code to make robot car go forward without diverging from the track and stop at thin black line.
- Code to make robot car stop if something is in the way.

```
void loop()
{
delay(50);
unsigned int uS = sonar.ping(); // Send ping, get ping time in microseconds (uS).

myservo.write(50);//goes to 50 degrees
delay(500);//wait for 0.5 seconds
myservo.write(110);//goes to 110 degrees
delay(100);//wait for 0.1 seconds

uint16_t r, g, b, c, colorTemp, lux;
int QRE_Left = analogRead(qreLeft);
int QRE_Right = analogRead(qreRight);
tcs.getRawData(&r, &g, &b, &c);

if ( uS / US_ROUNDTRIP_CM > 10 || uS / US_ROUNDTRIP_CM == 0)
{

if (r > 60) //if color sensor is on the bare floor
{
if (QRE_Left > threshold && QRE_Right > threshold)
{
RGB_color(0, 255, 0); // Green
forward();
}
else if (QRE_Left < threshold && QRE_Right > threshold)
{
RGB_color(0, 255, 0); // Green
bearRight();
}
else if (QRE_Left > threshold && QRE_Right < threshold)
{
RGB_color(0, 255, 0); // Green
bearLeft();
}
}
else //If color sensor is on the red tape marker
{
allStop();
RGB_color(255, 0, 0); // Red
delay(2000); //Stop as long as you want - in this case, 2 seconds
RGB_color(255, 255, 0); // Yellow
forward();
delay(100);
}
}
else if (uS / US_ROUNDTRIP_CM < 10)
{
RGB_color(0 , 0, 255); // Blue
allStop();
delay(1000);
}
}
```

Code

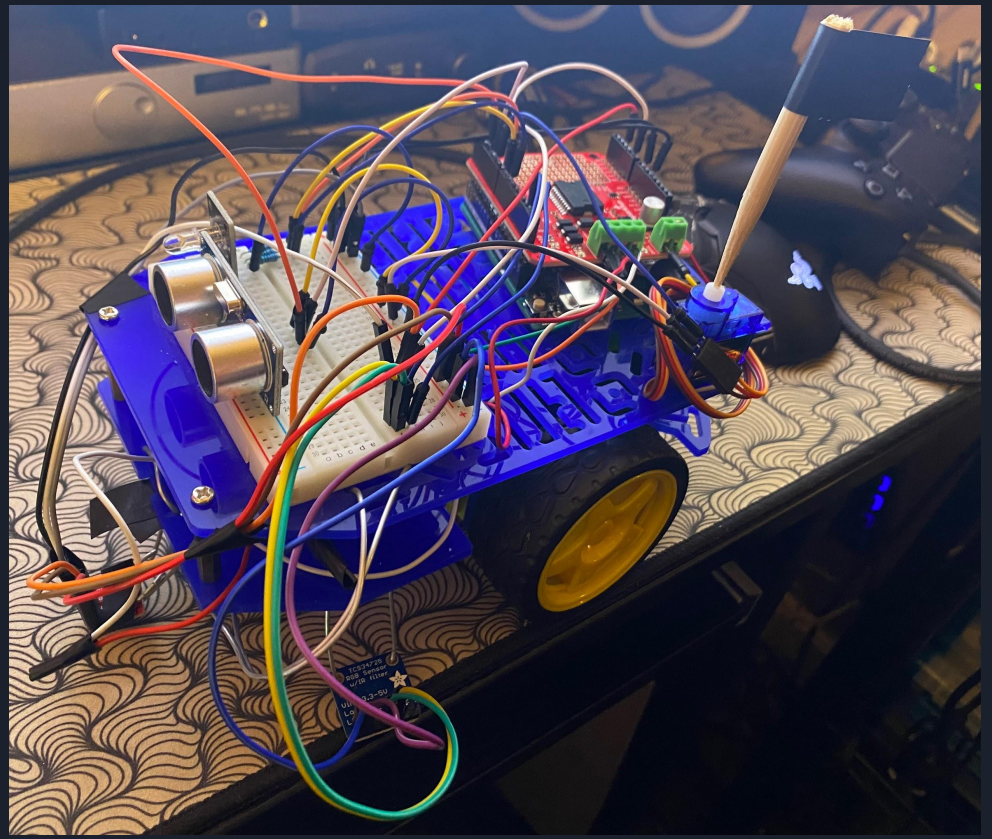
- Void function to make RGB led light up the correct color.
- Void function to set speed and direction of Motor A and Motor B.
- Void functions to steer robot forward, right, or left.
- Void functions to stop robot car.

```
//RGB led function
void RGB_color(int red_light_value, int green_light_value, int blue_light_value)
 {
  analogWrite(red_light_pin, red_light_value);
  analogWrite(green_light_pin, green_light_value);
  analogWrite(blue_light_pin, blue_light_value);
}
// driveArdumoto drives 'motor' in direction 'dir' at speed 'spd'
void driveArdumoto(byte motor, byte dir, byte spd)
{
if (motor == MOTOR_A)
{
digitalWrite(DIRA, dir);
analogWrite(PWMA, spd);
}
else if (motor == MOTOR_B)
{
digitalWrite(DIRB, dir);
analogWrite(PWMB, spd);
}
}
void forward() // Runs both motors at speed 'spd'
{
driveArdumoto(MOTOR_A, CW, spd); // Motor A at speed spd
driveArdumoto(MOTOR_B, CW, spd); // Motor B at speed spd
}
void bearRight()
{
driveArdumoto(MOTOR_B, CW, 0); //Motor B Stop
driveArdumoto(MOTOR_A, CW, hiSpd); //Motor A hiSpd
}
void bearLeft()
{
driveArdumoto(MOTOR_B, CW, hiSpd); //Motor B hiSpd
driveArdumoto(MOTOR_A, CW, 0); //Motor A Stop
}
// stopArdumoto makes a motor stop
void stopArdumoto(byte motor)
{
driveArdumoto(motor, 0, 0);
}
void allStop() //Stop both motors
{
  stopArdumoto(MOTOR_A); // Stop motor A
// All pins should be set up as outputs:
pinMode(PWMA, OUTPUT);
pinMode(PWMB, OUTPUT);
pinMode(DIRA, OUTPUT);
pinMode(DIRB, OUTPUT);

// Initialize all pins as low:
digitalWrite(PWMA, LOW);
digitalWrite(PWMB, LOW);
digitalWrite(DIRA, LOW);
digitalWrite(DIRB, LOW);
}
```

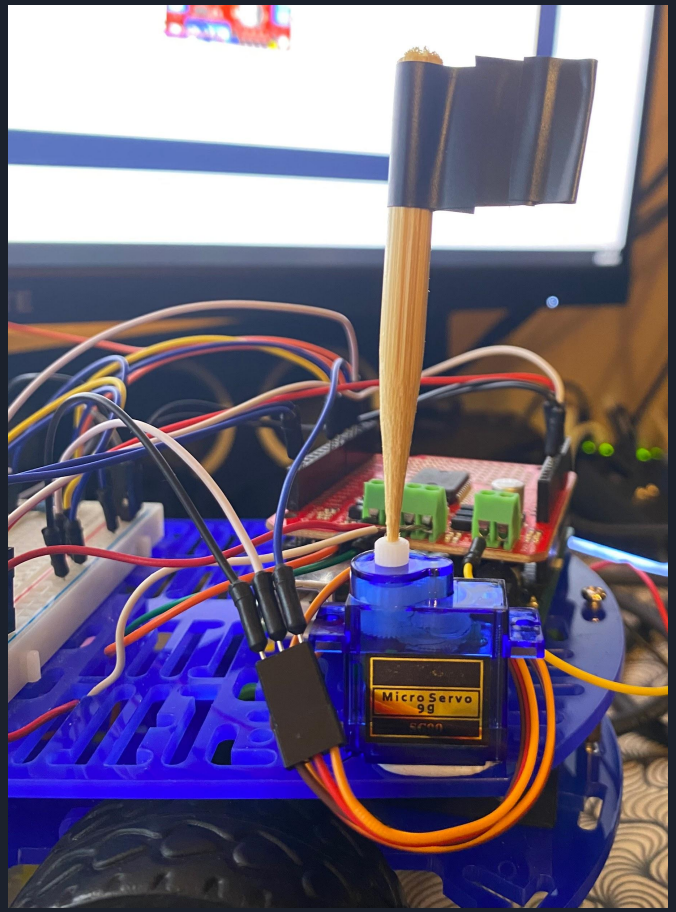
Robot Car

- Image of robot car completely built.



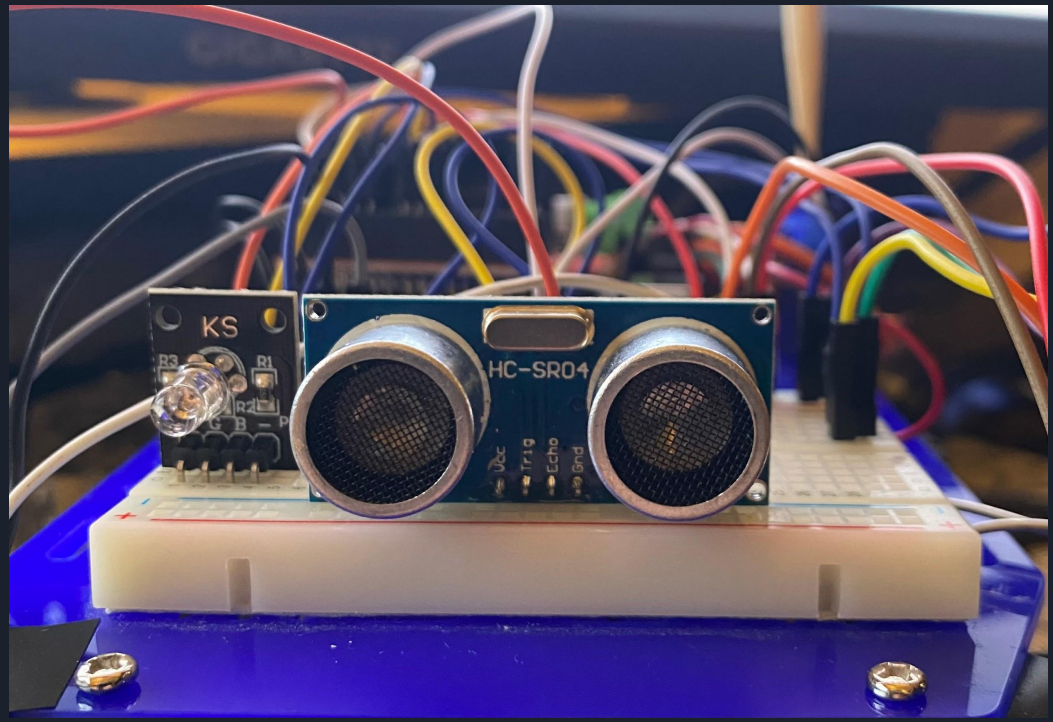
Servo Module

- Servo module makes flag spin to seem as if it is in the wind.



Ultrasonic Sensor Module

- Ultrasonic sensor makes sure robot car doesn't crash into anything.



RGB LED Module

- The RGB LED will light up different colors depending on the situation.

- Red - the robot car is stopped due to the color sensor.
- Blue - the robot car stopped due Ultrasonic detection.
- Yellow - the robot car is attempting to steer forward.
- Green - the robot car is moving straight forward.

Troubleshooting

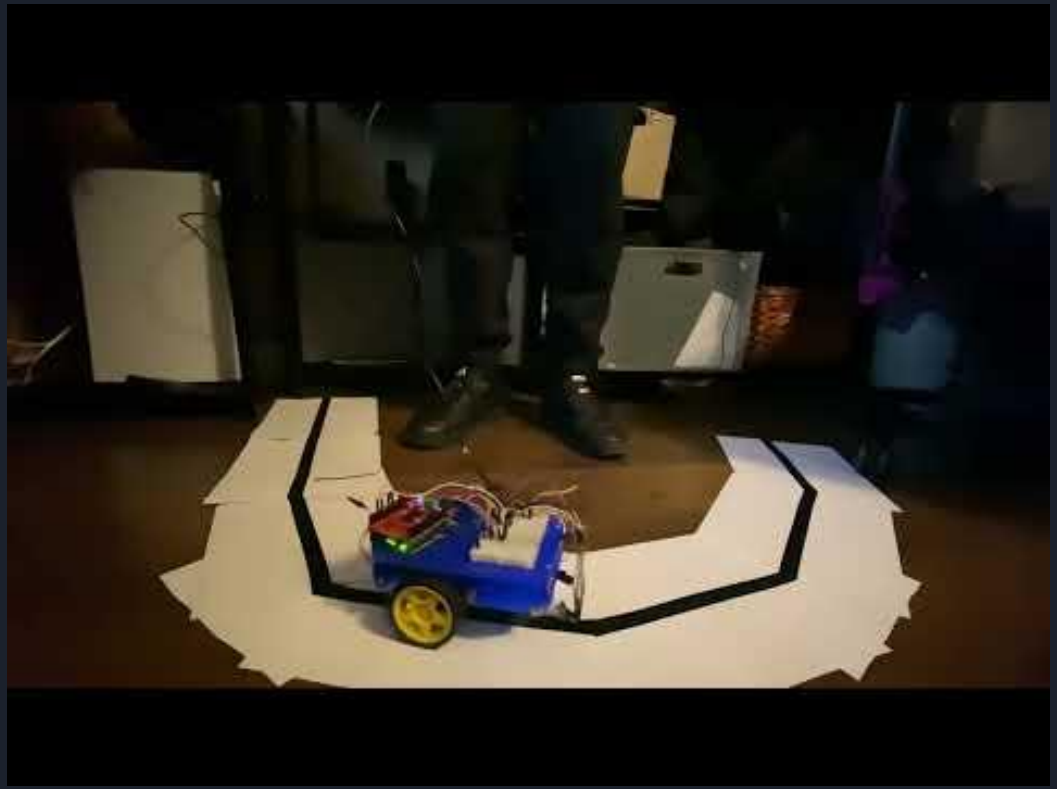
- Components such as the line sensor were very finicky
- Motors wouldn’t work properly.
- Arduino would sometimes work properly sometimes not.
- Powering the robot was the main issue we believe because it wouldn’t work correctly without being plugged to the computer using the usb cable.

Most of these issues were resolved with trial and error. The arduino would still sometimes ignore the code uploaded and the power required to run the motors on batteries alone didn’t work at all.

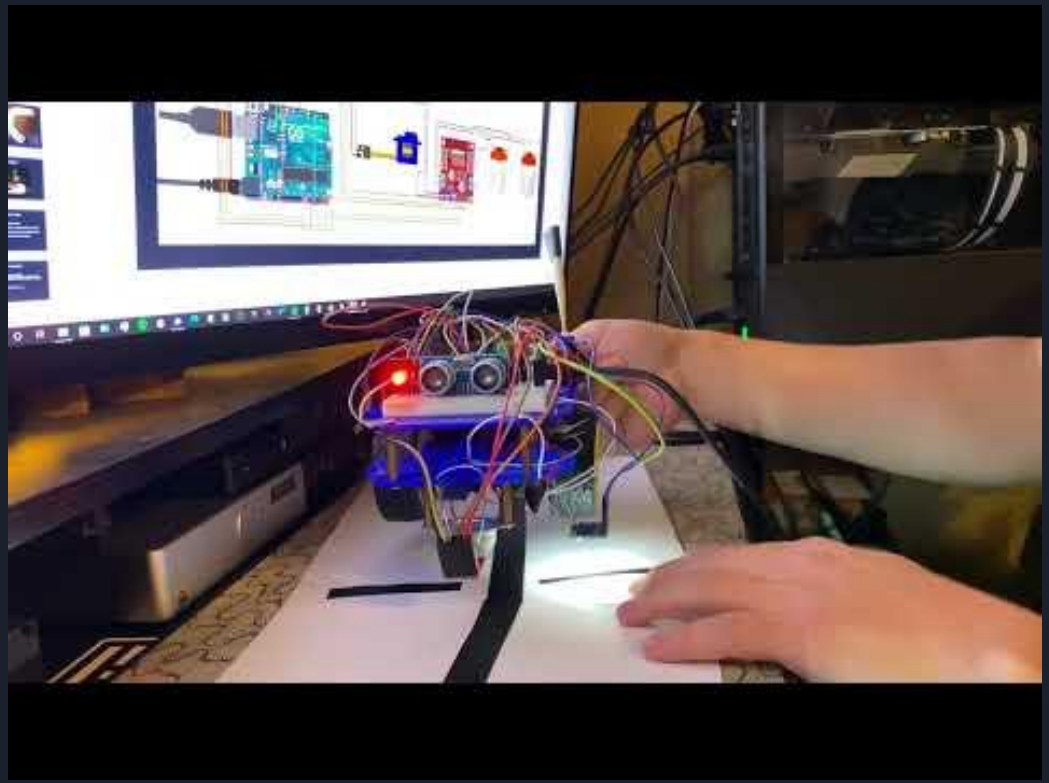
Track Setup

- The robot car must follow the black line and stop at the small black line.

Video Demo



Video Demo



How it works

- The arduino runs the code that will move the car forward
- Periodically checking for the black line
  - If the line is no longer detected by the LEFT sensor the arduino will move the car a bit to the right
  - If the line is no longer detected by the RIGHT sensor the arduino will move the car a bit to the left
- When the car detects the thin black line with the color sensor or an obstacle with the ultrasonic sensor, it will stop.
- The RGB led lights up a corresponding color depending on the robot cars status.
- The servo rotates a flag pole clockwise and counterclockwise to simulate a moving flag.

Conclusion

- The robot sensors were difficult to get to work properly.
- The motors did not have enough power to fully move the car forward, we did manage to get it to move itself faster later.
- However, everything else worked fine.
- Overall this project was insightful and fun.