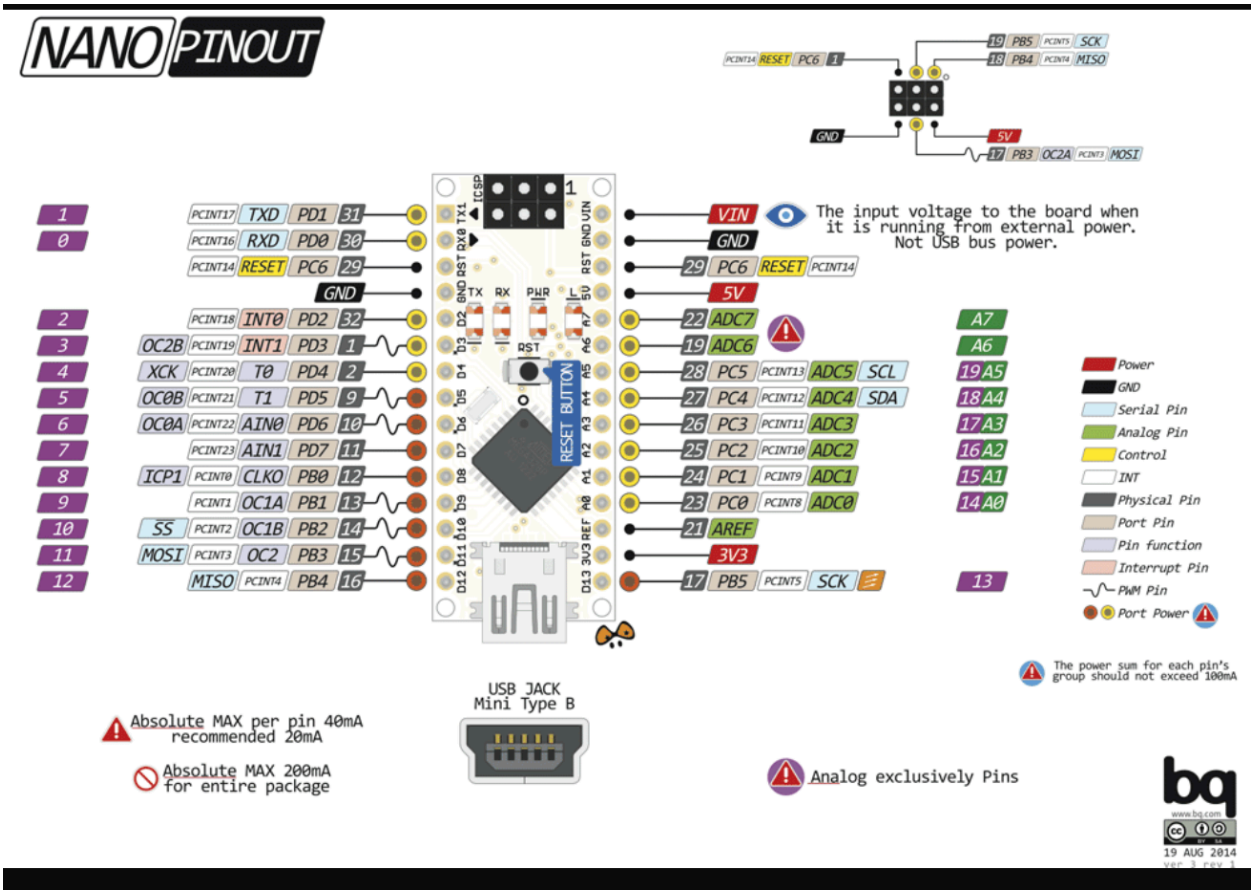
Devyan Biswas: 804988161 Enika Biswas: 004997956

INTRODUCTION AND CONCEPT(S)

The main goal of this project was to take concepts learned over the course of the quarter and apply them to a real-world product. In this case, it would be the IR controlled car. The main goal of the car itself is to follow a track and stop at the end, indicated by a black line. The track itself was a black line approx. .5-.7 inches in width on a white background. The car had to be able to go straight, perform two "lane changes", ie turns, and then stop at the end to the track, indicated by the black line mentioned above. Additionally, for every right turn it is to turn a red LED on, every left turn a blue, and if it moves straight a green. The fundamental concepts are as follows and will be expounded upon in the next few paragraphs: the Arduino Nano, photo-transistors, motors, IR LED's, and finally the concept of the guardrail approach to the maneuvering of the car.

The Arduino nano is a micro-controller/board that can interface with the Arduino Software, allowing for programmers to write programs on their own computers, upload to the Arduino, and have it run on boot(by default). Here is the pinout diagram for the Arduino Nano:



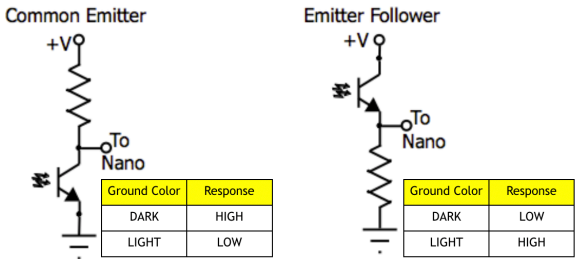
[1]

Devyan Biswas: 804988161 Enika Biswas: 004997956

The Nano has both digital and analog pins. However, some of the digital pins have a squiggle on them, which represents that they are also PWM pins, or Pulse Width Modulation pins. What PWM does is allow us to emulate a square wave(ie a wave that varies between on and off) with voltages between 5V and 0V by changing the portion of the time the signal spends on versus the time that the signal spends off[4]. The net effect is a variable power output(given the switching occurs fast enough), which is what the pulse width modulation gives us. The digital pins themselves allow us to send power, as do the analogs. However, for this project, we use the analog pins to read information from the IR sensors(IR LED with an IR phototransistor; range 0 to 1023), and the digital pins with PWM are used to control the motors, while the digital pins are used for the LED's. [5] To read information from the analog pins, we use the analogRead() function built into the C library for the Arduino Nano. This means that it will map input voltages between 0 and 5 volts into integer values between 0 and 1023[3]. We also use the digitalWrite() with the PWM pin for values between 0 and 255(low and 5V) and with the normal digital pins for either LOW or HIGH(0 or 5V).

The motors themselves are relatively standard. One thing to note***, is that we needed to flip our leads for our motors, however, that was simply because they were soldered on backwards. Otherwise, the motors take a direct voltage, generated by the PWM digital pins(as described above).

The IR unit is the main crux of the control system. The actual data from the sensor will be detailed in later sections. But here is a brief overview of the concept. Essentially, the IR LED acts as the emitter, and the IR phototransistor acts as the receiver. The IR phototransistor's resistance and output voltage, therefore, change in proportion to the IR light it picks up. The two ways you can set up the sensor is a common emitter system, with the system displaying high voltage when reflecting off of a dark surface and low off of a white surface, or an emitter follower system, with the opposite response. The difference is where we are measuring. For the common emitter, we are reading the analog values before the phototransistor and after the resistor(which is coming from the 5V source on the Nano). And for the emitter follower system, we read the values after the phototransistor(coming from the 5V source on the Nano) and before the resistor.[2]



[FROM LECTURE SLIDES]

The last topic integrates these all together: the system by which you actually adjust the motor speed. The method we opted for was the guardrail method, which was a much simpler task in implementation, but took some testing to specify the ranges for values for which to adjust motor speeds(more on this later in the testing section). Essentially, you can imagine this method as a bumper car approach. When the car goes too far to the left, it recognizes this and adjusts itself by speeding up the wheel that turns it towards the track and slowing down the opposite wheel, allowing for a quicker adjustment. In fact, this method would only require two IR sensors(one to detect left and one for right), however, adding a third central IR sensor introduces a bit more precision when it comes to the adjustment. Overall, the system works like this. It continuously loops and polls the 3 IR sensors for data, and when each sensor is above/below a certain threshold(mentioned above), it will perform accordingly. If the right sensor's value goes high, that means that it is over the black line, which means the car has to turn right to adjust for this. And when the left sensor is high, that means it is over the black line, meaning that it has to turn left to adjust for this. Once the center sensor is on the black line, it will go high. And so long as the other two sensors are below a certain range, the car will move straight. Once both the left and right sensors read above a certain threshold, we know they are both over a black line, and we stop the car.

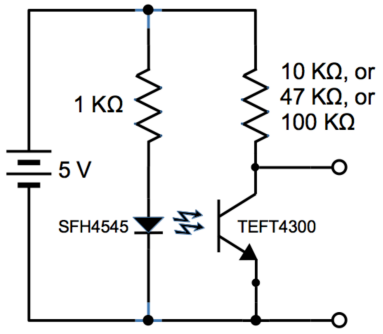
TESTING AND METHODOLOGIES

IR TESTS

IR Functionality

Test design

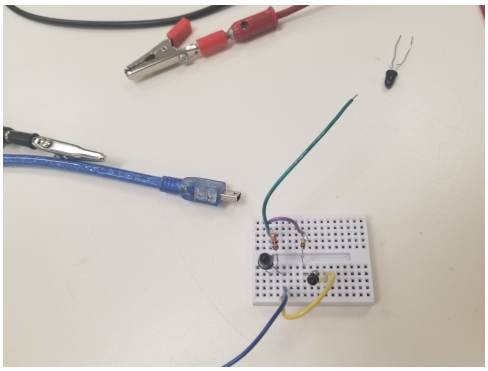
To test that the IR sensors work, we used the below circuit design with different values of resistors: 100K, 10K, and 47K ohm resistors and observed the values of voltage output of each. Below is the electrical circuit used:



[FROM PREVIOUS LAB]

The physical characteristics of this test was to simply populate a breadboard using the above schematic, and place the pins of the DMM across the IR LED. Below is a picture of this physical setup:

Devyan Biswas: 804988161 Enika Biswas: 004997956



How the test was conducted and Data Analysis

We used a DMM to measure the voltages across the IR's when there was no object covering the IR LED's and when we placed a white piece of paper over the IR LED's. Below were the results of the DMM reading:

	100 KOhm	47 KOhm	10 KOhm
Obj. Abv.	0.18V | 0.048mA	0.21V | 0.102mA	3.32V | 0.179mA
No Obj. Abv.	2.52V | 0.025mA	3.44V | 0.033mA	4.610V | 0.03mA

From this table, we can analyze the data and observe that the voltage is generally higher when there is no object above the IR LED. However, with white paper the voltage is significantly lower.

Data interpretation

From the above table values, we can conclude that the 47K ohm resistor is ideal since its voltage value is much more discernible, with not too tight a range but with values high enough to notice. Additionally, this test proved that the IR LED properly reacts when an object is placed over it as shown by the change in voltage.

IR Test with black vs white colors

Test design:

Next, we have to check the range of values from the IR sensor when it detects black or white colors. We did this test once we had decided on a physical orientation for the IR sensors(will be shown later below). The physical set up involved having a blank page with one vertical black line and then simply moving the car across the page horizontally. Then, we re-orient the paper so that the black line is horizontal and move the car vertically to test for the stopping condition when all three IR's are at the black line.

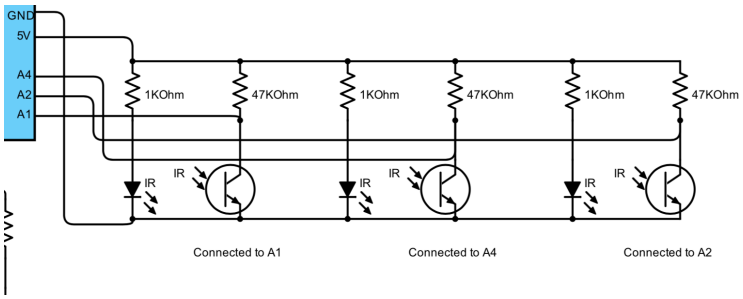
The electrical component of this test is to read the IR LED values when the car is moved across the page and moved vertically along the page by adding an analog read and print statement in the code like so:

Devyan Biswas: 804988161 **Enika Biswas: 004997956**

testingIR

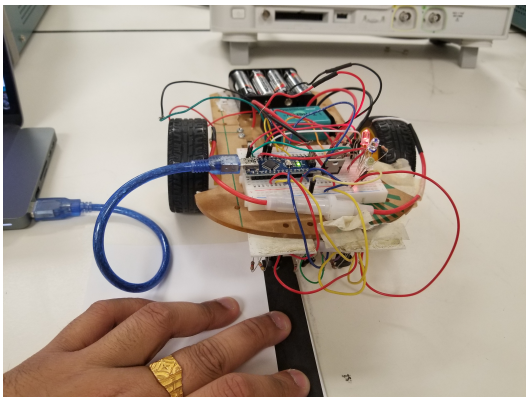
```
void setup()
{
  //LED
  pinMode(A4, INPUT);
  pinMode(A2, INPUT);
  pinMode(A1, INPUT);
  Serial.begin(9600);
}

void loop()
{
  Serial.print("IRLeft:");
  Serial.println(analogRead(A2));
  Serial.print("IRMid:");
  Serial.println(analogRead(A4));
  Serial.print("IRRight:");
  Serial.println(analogRead(A1));
  Serial.print('\n');
  delay(100);
}
```



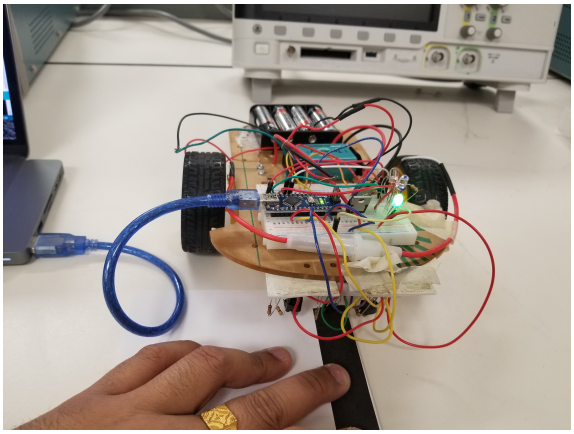
How the test was conducted and Data Analysis

When moving the car horizontally, we start by placing the car to the far left of the page so that the left and middle IR sensor is facing the white part of the paper. Below is the data of this test and the corresponding physical setup:



IRLeft	IRMid	IRRight
46	37	812
45	38	812
46	38	813
45	38	813
45	37	812
45	38	812
45	38	812
46	38	813
45	38	813
46	38	812
46	38	812
46	38	812

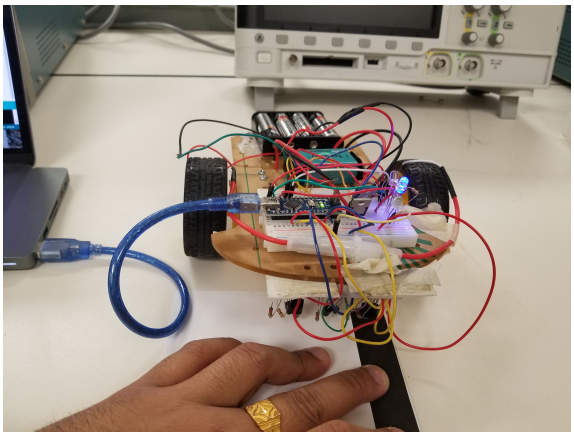
Next, we moved the car so that the middle IR sensor was facing the black portion. Below is the data of this test and the corresponding physical setup:



IRLeft	IRMid	IRRight
46	623	56
45	623	56
46	623	56
45	624	56
45	624	57
45	624	56
45	623	56
46	623	55
45	623	56
46	623	56
46	624	56
46	624	57

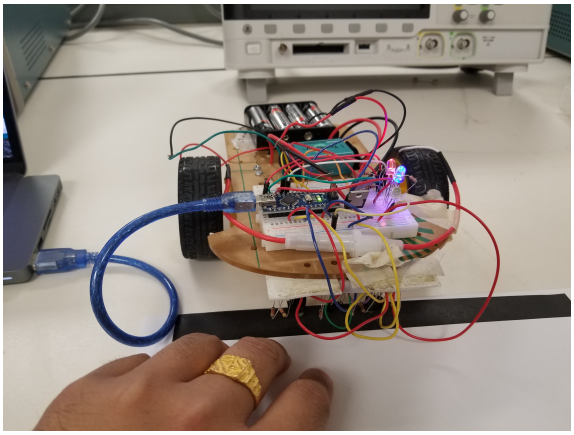
Devyan Biswas: 804988161 Enika Biswas: 004997956

We moved the car to the extreme right side of the paper, so that the right most and middle IR sensor were facing the white part of the paper. Below is the data of this test and the corresponding physical setup:



IRLeft	IRMid	IRRight
814	35	48
814	35	48
814	35	48
815	35	48
814	35	48
813	35	48
813	35	48
813	35	48
814	35	48
814	35	48
813	35	48
813	35	48

Lastly, we moved the car so that all sensors were over the black line. Below is the data of this test and the corresponding physical setup:



IRLeft	IRMid	IRRight
814	609	762
814	609	762
814	609	762
814	609	762
814	609	762
814	609	762
814	609	762
814	609	762
814	609	762
814	609	762
814	609	762
814	609	762

From these values, we can analyze the data and
observe that when the IR sensors hovered over the black portion of the page, the values read were higher and got close to 800. When the IR sensors hovered over the white portion of the page, the values reduced to somewhere below 50.

Data interpretation:

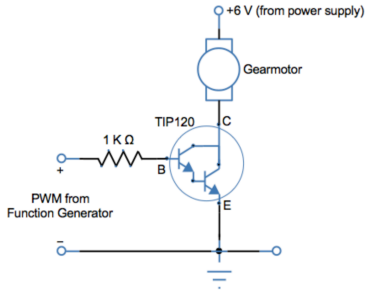
These test results and following data analysis tells us our range, which is somewhere between 600 to 800 when the IR sensors are above the black portion of the page. When the IR sensors are above the white portion of the page, the values always stay below 50 close to 40.

Devyan Biswas: 804988161 Enika Biswas: 004997956

MOTOR TEST

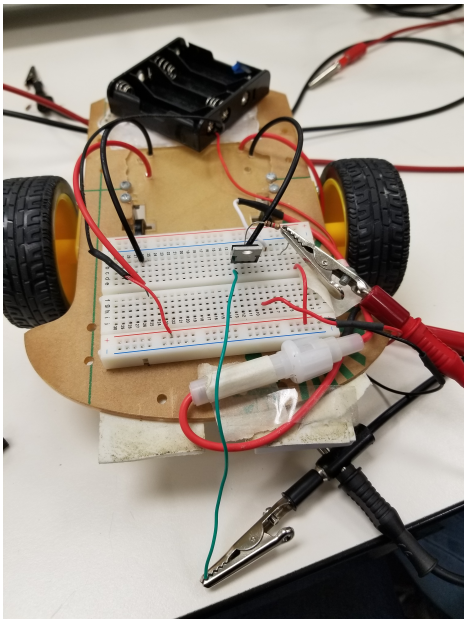
Test design:

To test the motor functionality we can see how increasing the duty cycle of the motor affects the motor's voltage and current readings as well as conclude the minimum duty cycle to start the motor. The electrical setup is given by the following schematic below:



[FROM PREVIOUS LAB]

The physical setup involves populating the breadboard given the above schematic and then connecting the circuit to the PWM of the function generator followed by connecting the function generator to the oscilloscope. Below is the physical setup:



How the test was conducted and Data analysis

Once the circuit is connected to the PWM of the function generator, we set up the generator to 6VPP and 500 Hz and connect the function generator to the oscilloscope.

Next we start with 10% duty cycle and read the average voltage of the PWM signal indicated on the oscilloscope screen, and the motor current indicated on the front panel of the power supply. Below are the results of the motor current and the average voltage with various duty cycles:

Devyan Biswas: 804988161 Enika Biswas: 004997956

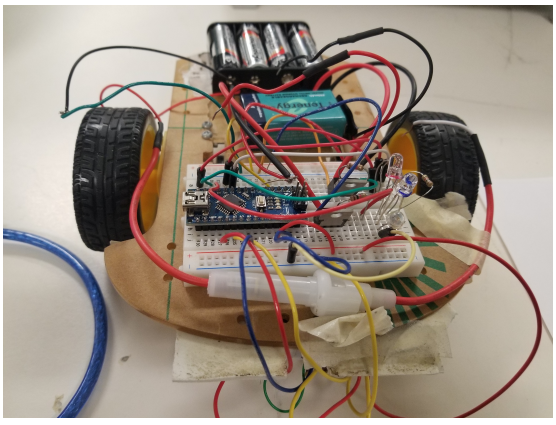
Duty Cycle: %	Avg. Motor: Volts	Mtr. Curr.: Amps
20	1.12	0.07
30	1.67	0.11
40	2.32	0.13
50	2.90	0.14
60	3.55	0.15
70	4.00	0.16
80	4.60	0.17
90	5.2	0.17
80	4.60	0.17
70	4.00	0.16
60	3.55	0.15
50	2.9	0.14
40	2.32	0.13
30	1.67	0.11
20	1.12	0.07
10	0.620	0.02

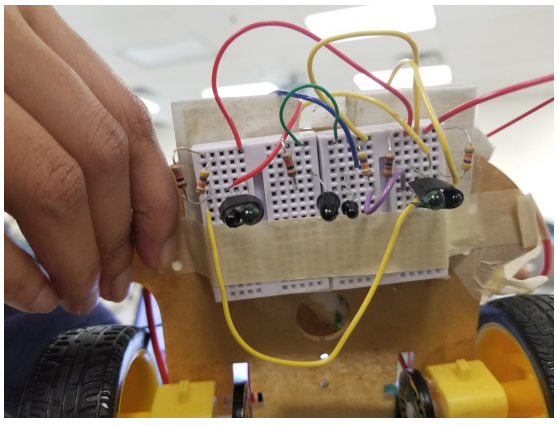
Data interpretation**:**

This test showed that with increasing duty cycle, the motor volts and current increase as well. Additionally, the minimum duty cycle to start the motor was 30% (note, that the speed was very slow at 40% duty cycle). This conclusion gives us insight into how fast the motor should be.

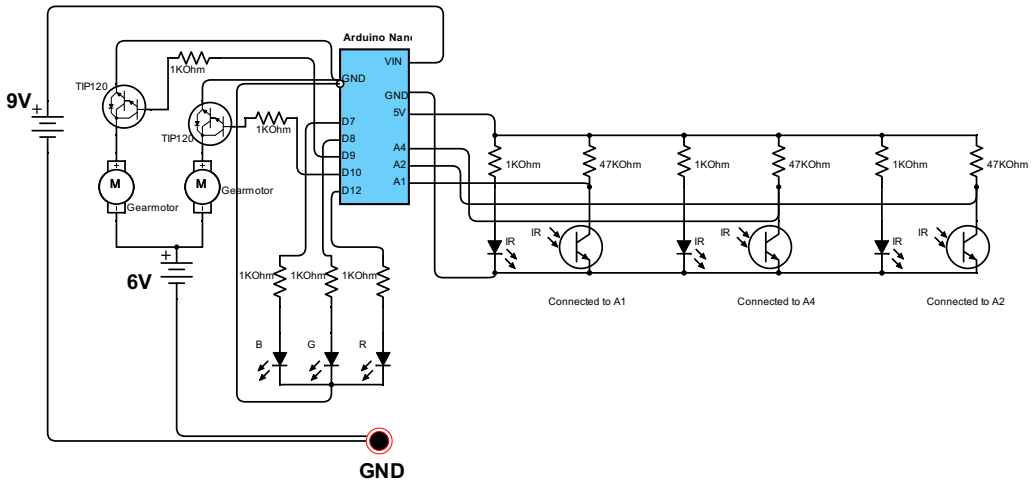
RESULTS & DISCUSSION

Using the above data, we have the details to formulate the software and integrate these 2 separate subsystems. Below are the pics of the fully assembled car (with all day-of electronics):





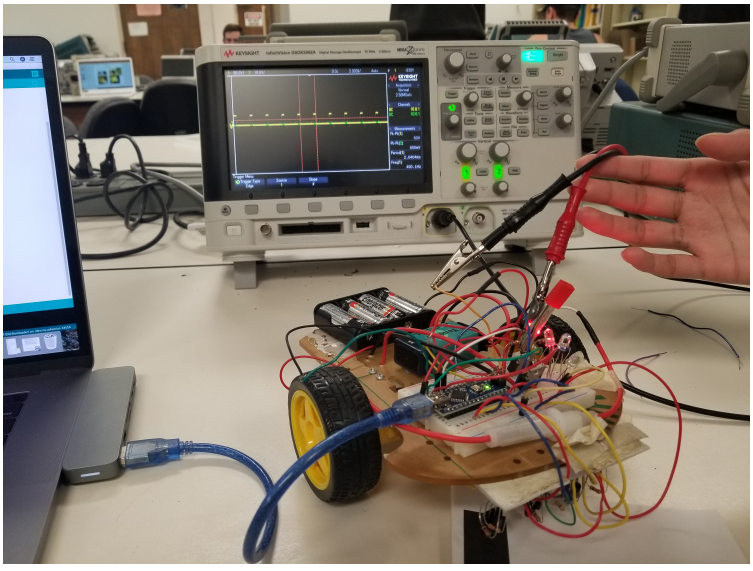
And here is the schematic:



[6]

Note: Battery for 9V is connected to the ground of the Nano, written as GND here simply for readability. Additionally, 6V is commented to its own ground. Also, the + and negative leads on our motors were flipped, but this was just due to a soldering issue, so disregard any discrepancy in the image fo the circuit vs. the schematic that arises due to that.

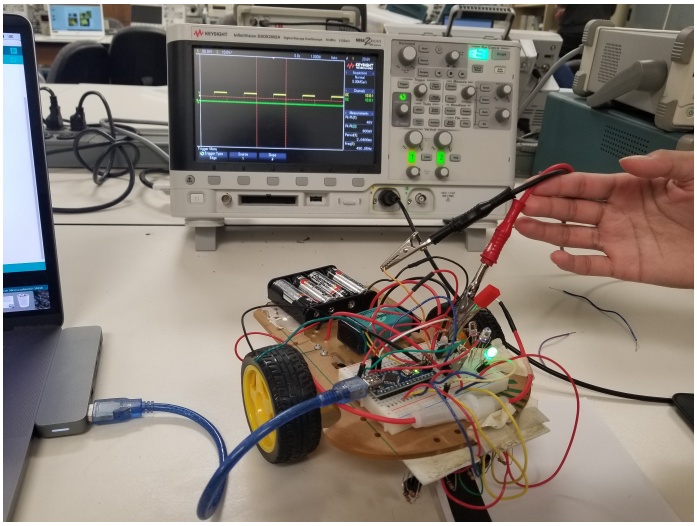
After integrating the two systems with the race car code (included in the zip file), we use an oscilloscope to check if we get expectant results. We had the scope across the left motor. (Note, because we did not scale particularly accurately, the number of square waves may look a bit out of scale):



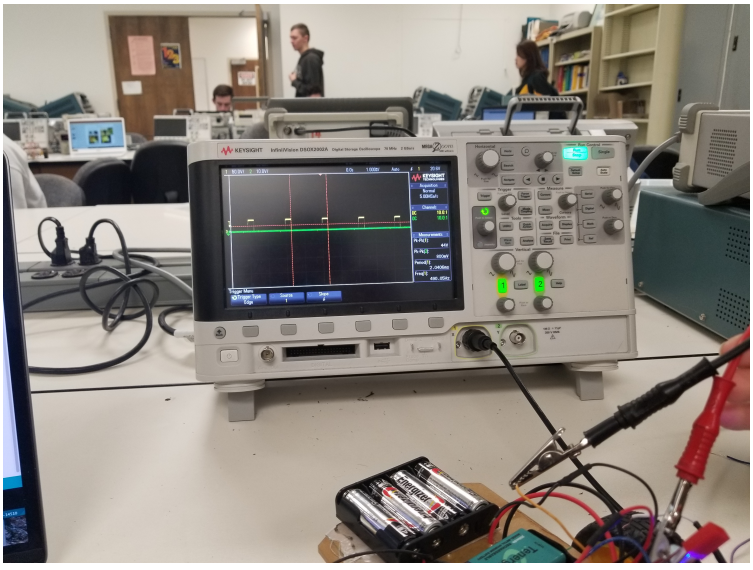
Devyan Biswas: 804988161 Enika Biswas: 004997956

At first, we start off in a situation where it should the car towards the right, meaning its left motor moves faster than the left motor. In the above image, the net voltage we are sending to the motor, like discussed in introductory concepts section, is approx. 5V.

As we go across, we see the value decreases across the left hand motor:



When the central sensor is above the black line, the car should be going straight, which is indicated by the left-motor now being at a lower voltage (4.6V).

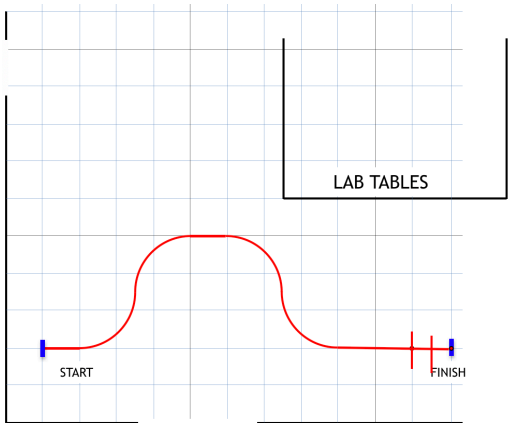


The blue light indicates that the left sensor is over the black line, so it has to adjust by turning left, meaning the left motor is now slower, which is shown by the scope (4.4V).

Devyan Biswas: 804988161 **Enika Biswas: 004997956**

IMPROVEMENTS

One of the things that we noted on race day was that our car was slow. It completed the track in about 29 seconds:



This could potentially be improved by fine-tuning the range so that the rover would not oscillate as much and as a result would not consume as much time. This could have been done by further testing with narrower ranges. Another improvement could be to use PID approach to also minimize the amount of oscillation.

PID is, in essence, a predictive speed control, and using this method would significantly smoothen the turning.

The P is the proportional value, which does not think about context, but only about adjustment to the error between expected and current (position):

$$h_r = LVS^* t_h$$
$$(error) = R - h_r$$

[SLIDES]

The D is the derivative control, which looks at rates of change in error and adjusts appropriately, is what makes this type of control predictive(velocity):

$$\frac{d}{dt}(error) = \frac{d}{dt}(R - h_r)$$

[SLIDES]

The I is integral controller, which looks at the internal of error. This essentially allows for a changing constant value for adjustment, sort of like a bridge between the proportional and derivative controllers, which increases the accuracy of the system:

$$K_i \int (error)dt = K_i \int (R - h_r)dt$$

CONCLUSION AND FURTHER WORK

Our goal was to make sure that the car reached the end of the track, which it did. Thus, out design did meet our main objective. Optimally, we hoped to make the car reach the end with minimal oscillations and minimal time. However, we did not have the time to perfect the range or try the PID approach in order to improve the time performance and minimize oscillations. Thus if we were to do this experiment again, we would improve the experiment by perfecting the range and trying the PID approach as well. We could perfect the range by narrowing the range and testing the car on the track again with these changes. The other approach is to change the software to implement PID techniques rather than guard rail.

We learned the proper way to approach a project and how to break up the work among team members to ensure that everyone is involved. One of the main values we learned is the value of testing. It is important to spend most of the time testing as this is where a workable product becomes improved and more efficient.

ILLUSTRATIONS AND LINK REFERENCES

References:

[1] https://bigdanzblog.wordpress.com/2015/01/30/cant-get-i2c-to-work-on-an-arduino-nano-pinout-diagrams/
[2] https://electronicsforu.com/resources/learn-electronics/ir-led-infrared-led-infrared-sensor
[3] https://www.arduino.cc/reference/en/#functions
[4] https://www.arduino.cc/en/Tutorial/PWM
[5] https://www.arduino.cc/en/Tutorial/Foundations
[6] SchemeIt software from Digikey: https://www.digikey.com/schemeit/project/carschematic-KMQLIBO4003G/