ECE 3 Final Report

by Utkarsh Kumar and Avnish Sengupta
ECE 3 Lab 1B

Introduction

In the final project of the Introduction to Electrical Engineering class, we were required to use the knowledge obtained from the course of using the TI Robotics System Learning Kit MAX [1] o assemble a robotic car capable of following a track provided to all students of the course. Specifically, the primary objective of this project was for our group's robotic car to successfully follow the provided curved track, perform a donut at the end of the track, and return and halt at the starting position for all four positions. Furthermore, we sought to achieve the aforementioned task in under 11 seconds.

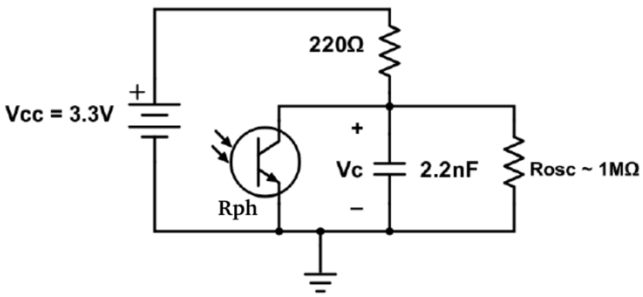
The two main goals of the project (getting the car to successfully complete the track from all four positions, and do so under 11 seconds) were accomplished via an implementation of the PID control system (described in the subsequent sections) and usage of the Energia IDE [2] to upload the path-following code onto the MSP432 LaunchPad microcontroller installed onto the car. The next section discusses in detail how the path-sensing system works.

Background

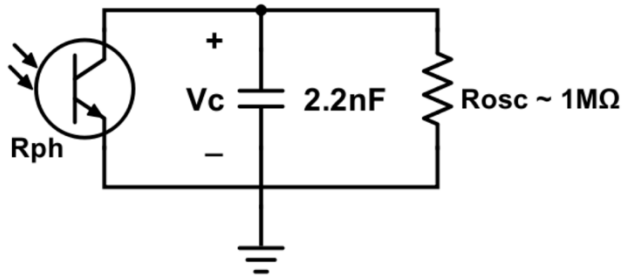
This section explains the basic operation and working principles of the path-sensing circuitry and the infrared sensor of this car. An explanation of the calibration and sensor-fusion process is also provided to give the reader a stronger understanding of how the car converts the sensor reading into steering commands.

Path-sensing circuitry:

Inside the car, there exists circuits consisting of resistors, capacitors, and phototransistors amongst many other components, which allow the car to collect IR sensor values. These values are unitless but give a measurement of how 'light' or 'dark' the ground below the car is at the instant of the data collection. To aid the understanding of such a circuit, a simplified version is given below and explained. The same principles would apply to the more complicated circuitry inside of the robotic car.



Charging stage (taken from EE3 Lab Manual Figure 3-13)



Discharging stage (taken from EE3 Lab Manual Figure 3-14)

Figure 1: The phototransistor circuit stages showing the two main cycles in a path-sensing circuit

Figure 1 shows the two stages of the phototransistor circuit similar to the one included in the car. This circuit consists of a charging and discharging stage where the time taken for the capacitor to fully discharge is measured and used to determine the darkness of the surface. The charging stage where the capacitor is charged from 0 volts to 3.3V (i.e. the Vcc = Vc). This stage is used between each reading to charge the capacitor back up before the next reading. The charging stage takes approximately 0.4 microseconds and thus the sampling rate is 7 milliseconds/sample such that sufficient time is given for the circuit to recharge. The discharging stage is where the capacitor is discharged from 3.3V to 1V and this time is measured as the "settling" time such that it is 5τ. Using knowledge from the 1st order circuit and the fact that a smaller resistance dominates in parallel connections, the time constant is derived to be as follows: $\tau = CR_{ph}$ where R_{ph} is the resistance of the phototransistor.

Treating the NPN BJT phototransistor as a variable resistor, the value of R_{ph} decreases as more light is measured and increases as the sensors detect more darkness. Considering the 8 sensors below the robotic cars are fitted with such phototransistors, a white surface will give a lower R_{ph} which gives a lower time constant and thus a lower settling time, vice versa for a black surface. Figure 2 illustrates the difference in voltage vs. time curves for different surfaces. Here the differences in steepness can be thought of as representing the differences in settling times (a steeper slope indicating a faster settling time and a smaller time constant).

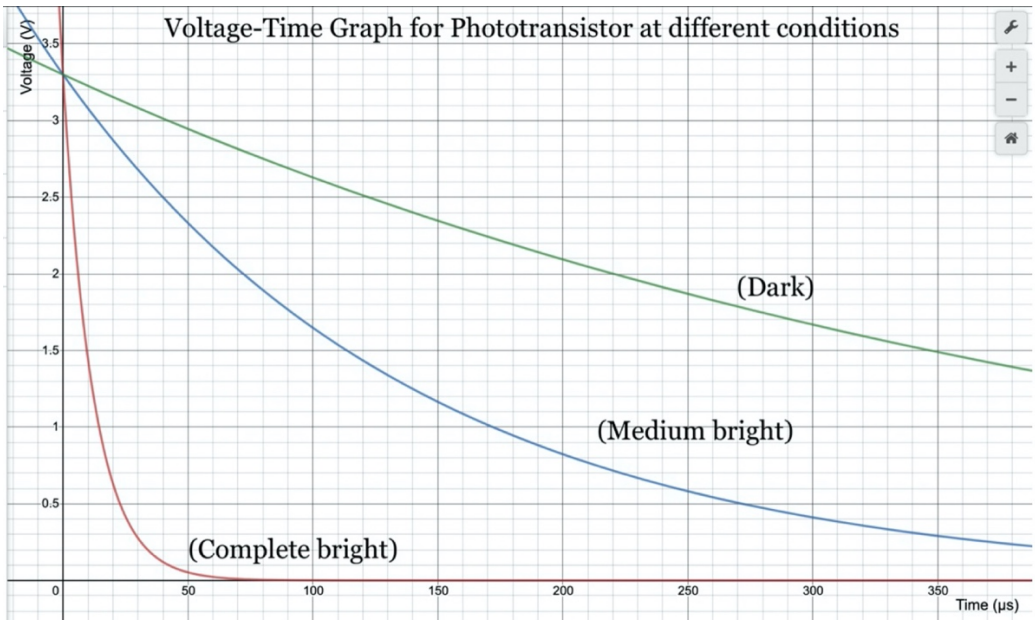


Figure 2: Discharge voltage vs. time curves for different conditions generated on Desmos [3]

The theory allows a differentiation of dark and light surfaces via measurements of settling times for each sensor on the bottom of the car. The settling time is scaled into a reading from 0 (white surface) to 2500 (dark surface) outputted for each sensor by the code provided in Figure 3. These values are used subsequently for the sensor calibration and fusion processes.

```
#include <ECE3.h>

uint16_t sensorValues[8];

void setup()
{
  ECE3_Init();
  Serial.begin(9600); // set the data rate in bits per second for serial data transmission
  delay(1000);
}


void loop()
{
  // read raw sensor values
  ECE3_read_IR(sensorValues);

  // print the sensor values as numbers from 0 to 2500,
  // where 0 means maximum reflectance and
  // 2500 means minimum reflectance
  for (unsigned char i = 0; i < 8; i++)
  {
    Serial.print(sensorValues[i]);
    Serial.print('\t'); // tab to format the raw data into columns in the Serial monitor
  }
  Serial.println();

  delay(1000);
}
```

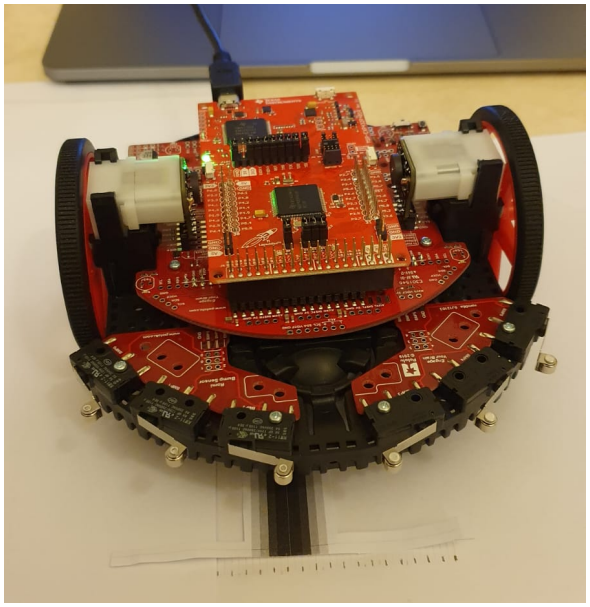
Figure 3: The code used to measure sensor values where 1 represents the rightmost sensor and 8 represents the leftmost sensor. The code was provided on CCLE.

Calibration process:

The calibration process implemented the code given in Figure 3 and Excel software [4] to generate plots for when each sensor reads the maximum sensor value (2500). The methodology of this process is briefly described below and the results are also given.

The procedure involved using a thin strip of calibration paper (black line at the center surrounded by lighter gray lines) and moving it from the rightmost to the leftmost sensors in 4 mm increments for a total of 20 times. The code from Figure 3 was uploaded to the microcontroller on the robot car and 5 readings were taken at each increment. The methodology for reading the data is shown in this video by Dr. Briggs [5]. The left most position (placed at the right-most sensor) was labelled -40 and increased by 4 mm until 40 at the right most position (placed at the left-most sensor). Figure 4 shows the picture of the setup used for the calibration process with the robotic car connected to the laptop whose screen displays the serial monitor on Energia printing an array of 8 sensor values. Additionally, the calibration was done carefully during night-time conditions (simulating the time at which race day will be conducted) and under household lighting to avoid the infrared rays from sunlight interfering. The calibration process also helped measure the minimum and maximum value each sensor reads which was crucial for sensor fusion.

The raw data collected from the serial monitor was pasted onto an excel sheet and the average of the 5 measurements of each sensor at each position was taken. A raw data table was then created containing these averages as provided below in figure 5.



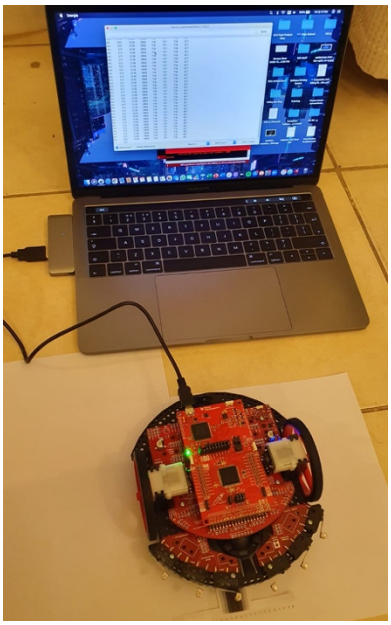


Figure 4: Calibration paper with the increments labelled on the white paper (left) A setup of the calibration measurement taken (right).

ORIGINAL DATA (AVERAGE OF 5 READINGS)								
ERROR	S1	S2	S3	S4	S5	S6	S7	S8
-40.00	2422.00	551.00	831.00	736.00	689.00	736.00	689.00	807.00
-36.00	2500.00	620.00	854.00	736.00	689.00	736.00	689.00	807.00
-32.00	2500.00	895.20	852.00	735.00	689.00	735.00	689.00	805.00
-28.00	1017.00	2038.00	1115.80	758.00	689.00	735.00	689.00	805.00
-24.00	946.00	2037.00	1333.60	758.00	707.40	735.00	684.40	805.00
-20.00	923.00	994.00	2500.00	758.00	707.40	735.00	684.40	805.00
-16.00	900.00	597.00	2500.00	1160.00	712.60	736.00	675.20	806.00
-12.00	900.00	574.00	2500.00	1728.60	713.00	736.00	689.00	806.00
-8.00	900.00	574.00	1349.00	2275.00	736.00	759.00	689.00	806.00
-4.00	899.00	574.00	941.40	2395.00	993.40	759.00	689.00	805.00
0.00	899.00	574.00	876.00	1466.00	2131.00	759.00	689.00	805.00
4.00	899.00	551.00	852.00	970.00	2371.00	852.00	689.00	805.00
8.00	899.20	551.00	852.00	782.00	2154.00	1679.00	689.20	805.20
12.00	899.00	551.00	852.00	758.00	899.00	2500.00	800.40	805.00
16.00	896.00	552.60	849.00	736.80	732.20	2500.00	1302.80	826.00
20.00	899.00	551.00	852.00	736.00	712.00	1325.00	2370.00	806.00
24.00	899.00	551.00	852.00	736.00	712.00	829.00	2500.00	1136.00
28.00	900.00	551.00	829.00	736.00	689.00	759.00	2058.00	2248.00
32.00	900.00	551.00	829.00	736.00	689.00	759.00	1066.00	2500.00
36.00	901.00	551.00	830.00	736.00	689.00	736.00	713.00	2500.00
40.00	901	551	830	736	689	736	689	1614
Minimum	896.00	551.00	829.00	735.00	689.00	735.00	675.20	805.00

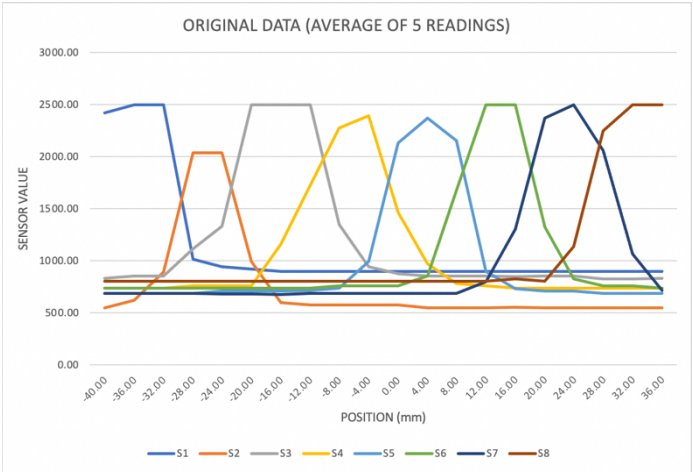


Figure 5a: Raw Data Table for Sensor Calibration

5b: Plot of raw mean sensor value vs. position

When the data is plotted as shown in Figure 5b, it is evident that each sensor has different minimum values and measure different maximum values throughout the calibration process. To prepare this data for the sensor fusion process (i.e. before it can be converted into steering commands), the minimum causing each curve to be offset needed to be removed such that the minimum error value was zero for each sensor and the data was scaled such that the maximum value was 1000. This allowed each sensor to have equal weighting in its error which is crucial when creating weighing schemes in the sensor fusion process. The normalization was done by dividing each column of sensor values by the maximum value and then multiplying by a 1000. Figure 6 shows the scaled data.

DATA NORMALIZED TO 1000 (DATA SCALING)

ERROR	S1	S2	S3	S4	S5	S6	S7	S8
-40.00	951.37	0.00	1.20	0.60	0.00	0.57	7.56	1.18
-36.00	1000.00	46.40	14.96	0.60	0.00	0.57	7.56	1.18
-32.00	1000.00	231.47	13.76	0.00	0.00	0.00	7.56	0.00
-28.00	75.44	1000.00	171.63	13.86	0.00	0.00	7.56	0.00
-24.00	31.17	999.33	301.97	13.86	10.94	0.00	5.04	0.00
-20.00	16.83	297.92	1000.00	13.86	10.94	0.00	5.04	0.00
-16.00	2.49	30.93	1000.00	256.02	14.03	0.57	0.00	0.59
-12.00	2.49	15.47	1000.00	598.55	14.27	0.57	7.56	0.59
-8.00	2.49	15.47	311.19	927.71	27.94	13.60	7.56	0.59
-4.00	1.87	15.47	67.27	1000.00	180.98	13.60	7.56	0.00
0.00	1.87	15.47	28.13	440.36	857.31	13.60	7.56	0.00
4.00	1.87	0.00	13.76	141.57	1000.00	66.29	7.56	0.00
8.00	2.00	0.00	13.76	28.31	870.99	534.84	7.67	0.12
12.00	1.87	0.00	13.76	13.86	124.85	1000.00	68.61	0.00
16.00	0.00	1.08	11.97	1.08	25.68	1000.00	343.93	12.39
20.00	1.87	0.00	13.76	0.60	13.67	334.28	928.76	0.59
24.00	1.87	0.00	13.76	0.60	13.67	53.26	1000.00	195.28
28.00	2.49	0.00	0.00	0.60	0.00	13.60	757.78	851.33
32.00	2.49	0.00	0.00	0.60	0.00	13.60	214.16	1000.00
36.00	3.12	0.00	0.60	0.60	0.00	0.57	20.71	1000.00
40.00	3.12	0.00	0.60	0.60	0.00	0.57	7.56	477.29

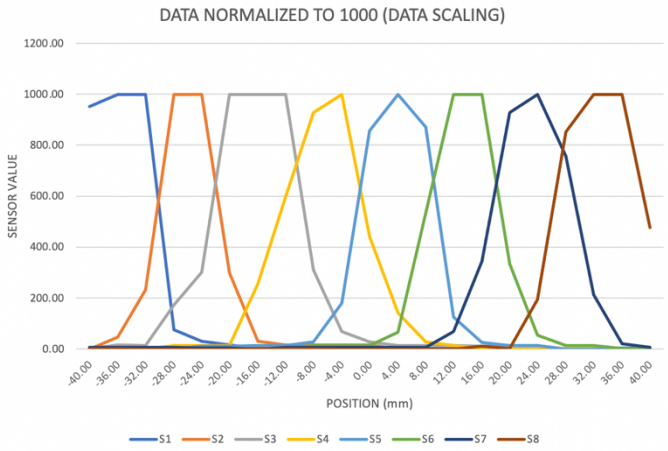


Figure 6a: Normalized Data Table for Sensor Calibration

6b: Plot of normalized sensor value vs. position

Sensor fusion process:

Once the data had been fully processed by the code in a similar fashion to the processing done via Excel, it needs to be converted into a steering command. In order to do so, the error values from each sensor at an instant in time must be converted into a single value. This may be done via several weighting schemes where each sensor is given a weighted value which increase/decreases with the distance of the sensor from the center. This is different from a guardrail implementation where the car would only respond when it is fully off-track. This form of sensor fusion allows the car to make small movements throughout to stay on-track and use all of its sensors appropriately.

The two weighing schemes used in Excel were (8-4-2-1)/4 and (15-14-12-8)/8. A 15-14-12-8)/8 scheme means the innermost sensors' error values are multiplied by 8 (i.e. sensors 4 and 5), sensors 3 and 6 are given a weighting of 12, sensors 2 and 7 a weighting of 14 and sensors 1 and 8 a weighting of 15. The total error value calculated by the weighted sum is then divided by 8 to obtain the final error value. The same process was carried out for (8-4-2-1)/4. The sensor fusion values using the two weighing schemes were plotted for different positions given in figure 7.

Looking at figure 7, it is evident that the (8-4-2-1)/4 weighting scheme is more linear in nature and gives constant commands for when the car is near the track but a relatively high error value once the car is towards the edge of the track. On the other hand, the (15-14-12-8) weighing scheme gives a large error reading even for slight fluctuations off the center of the track and stabilizes at greater distances away from the track. These two weighing schemes thus behave in distinct ways and determining which scheme will work best for accomplishing the project goals' was a consideration when performing the tests on the car (discussed in the following section).

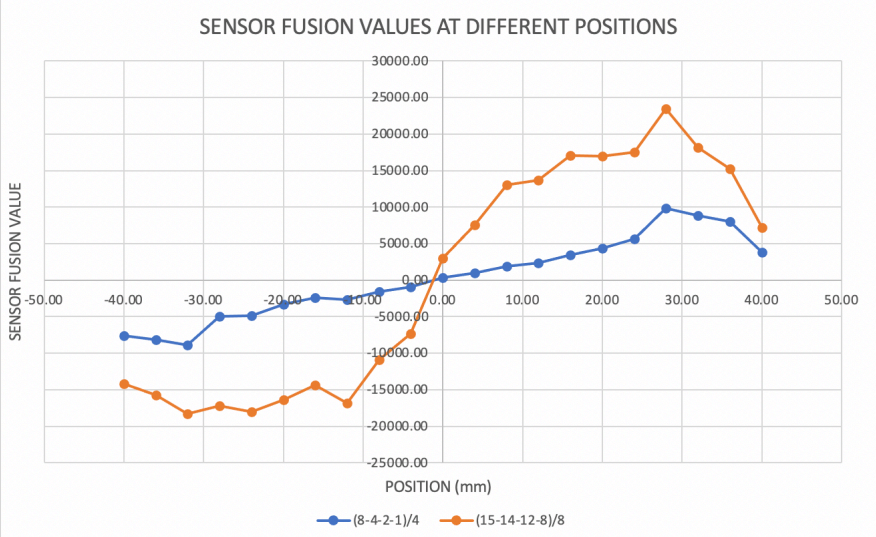


Figure 7: Line plot displaying the sensor fusion values for two different weighting schemes at different positions.

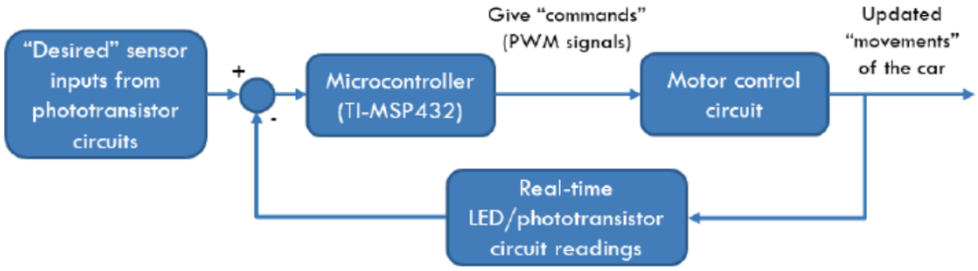


Figure 8: A high-level understanding of the final project (taken from EE3 Lab Manual Figure 3-1)

To summarize, figure 8 shows what the closed-loop feedback system for the line-following robot car looks like [6]. The path-sensing circuitry is used to obtain sensor input values for the car which is processed in the code uploaded to the microcontroller. The processing done in the code is identical to the sensor calibration and fusion discussed previously. The obtained scaled error values are then used in conjunction with proportional and derivative controls, discussed subsequently, to give PWM signals to the car. This mechanism is explained well by Electronics Tutorials [7]: "works by driving the motor with a series of "ON-OFF" pulses and varying the duty cycle, the fraction of time that the output voltage is "ON" compared to when it is "OFF." Figure 9 aids in the understanding of a PWM signal. Regarding our microcontroller, the PWM signal varied between 0 and 255 where 255 represented 100% duty cycle (i.e. 3.3V).

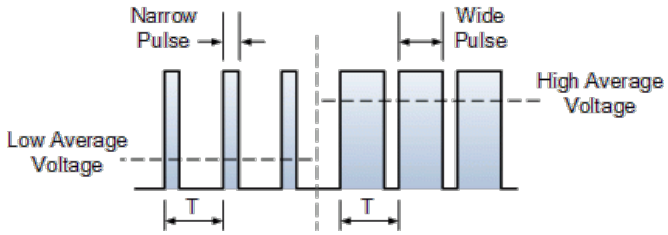


Figure 9: Key features of a PWM signal

Development Methodology

Test Setup:

Race Day was in the evening of our time zones – IST and GST – hence it was imperative that testing be done under similar conditions due to the sensitivity of the IR sensors on the robot car. The two main considerations were as follows: the ground on which the track was taped and the lighting of the room in which the tests were conducted. To ensure minimal interference of the ground with the IR sensors, additional white paper was added below and next to the track. Furthermore, both partners closed the windows and curtains in their room to avoid sunlight from interfering with the measurements and instead household and LED lighting was turned on.

The straight-line and curved tracks were printed on A4 pieces of paper and taped together on their sides and onto the ground to ensure they were intact. Figure 10b shows the two fully constructed tracks side-by-side and 10c shows a close-up image showing precisely how the pieces of paper were taped together. No tape was put on the track itself to avoid the sensors from picking up the tape's reflection. Furthermore, tape was put underneath the track as well to ensure flatness of the track. Once the tracks were prepared, the code was uploaded onto the TI-RSLK and then the car was placed onto one of the four starting positions. The setup for this is shown in figure 10a. Regarding the electrical characteristics of the setup, an AA, 1.5V energizer battery was used—picture provided in figure 11.

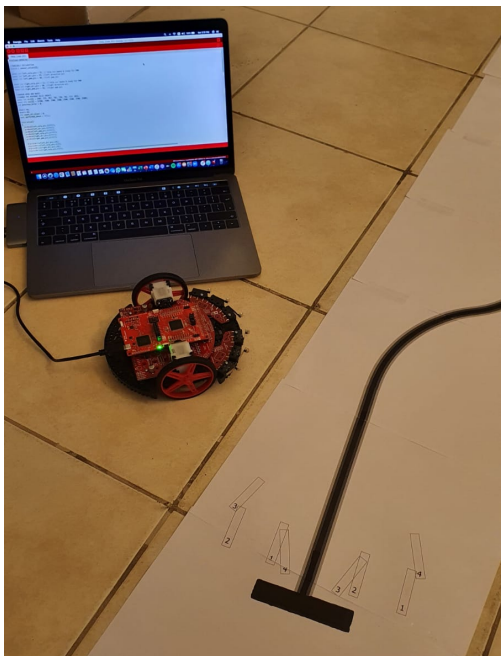
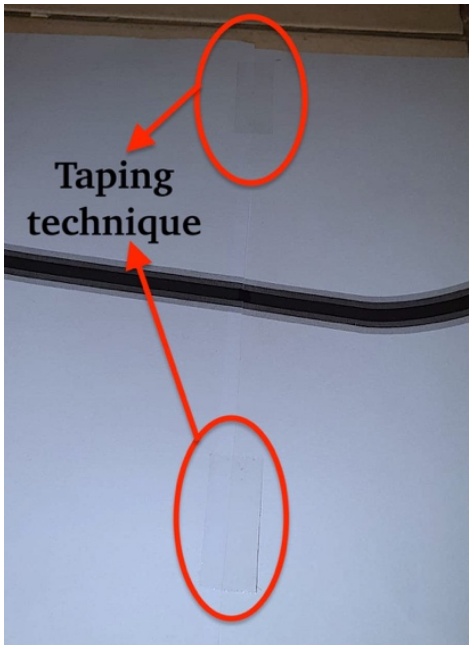


Figure 10a: Setup of the test run

10b: Fully constructed tracks (curved and straight)



10c: Close-up image of the track's construction

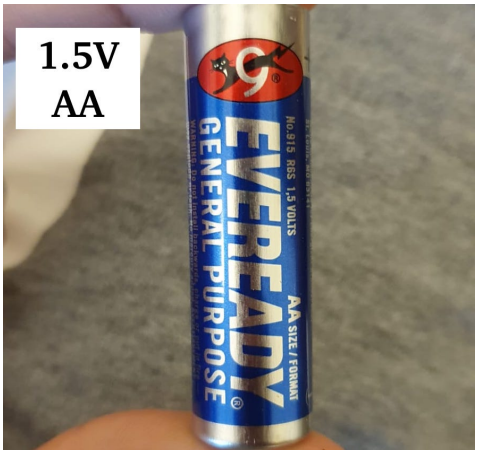


Figure 11: Battery used for testing

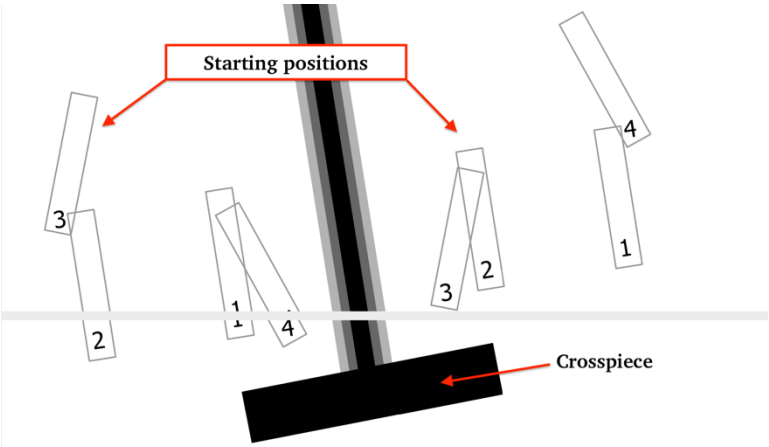


Figure 12: The starting of the curved track

Method:

The general methodology for carrying out each test was as follows:

1. Take a reading of the sensor values on a white piece of paper placed on the ground.
2. Replace the minimum sensor values array in the code with these new measurements to account for change in lighting conditions.
3. Set the desired values in the code for the parameters (Kp, Kd, base speed, acceleration, etc.)
4. Verify the code and then upload it onto the microcontroller as shown in Figure 10a, don't turn the robot car on yet.
5. Place the robotic car onto the chosen starting position shown in Figure 12.
6. Once in position, turn the car on and it should start following the track after a 1 second delay, wait for the car to reach the crosspiece on the other end and return back to the starting crosspiece. Be prepared for the car to go off track.

These six steps were followed when determining the value for each of the parameters involved in the code of the car. The key parameters were Kp, Kd, base speed, and acceleration. Other factors taken into consideration were the starting positions and qualitative observations such as reasons why the car failed to complete the lap or the amount of 'hunting' (i.e. oscillating around the track) it would do with different combinations of Kp and Kd. Notice that the serial monitor was not used when the car was on the track, it was only used when determining the minimum sensor values. Please refer to the code document to see detailed explanations for each section of the code and where each parameter is implemented as a variable.

Before discussing the methodology used to find the values of Kp and Kd specifically, a brief explanation of how the PID controls work is required [8]. When the car is off the track by a certain amount, say to the left, it is assigned an error value as shown via the sensor fusion process. This error value is negative when the right-side sensors see a dark surface implying that the error is negative when the car is to the left of the track. This requires a steering command to the right by a certain amount which is determined by the Kp value, this is called a proportional

controller. When the car's error, however, is measured over time, a steering command is given based on the direction in which the error is changing. This steering command is given by multiplying the change in error with a Kd value (derivative constant), this is called a derivative controller. Using the proportional and derivative controllers in conjunction with each other, a composite steering command is sent to the car. Kp and Kd are thus crucial parameters to determining not only how smoothly and stably the car follows the track, but also whether it successfully does so without going off track.

Data Analysis:

When determining the Kp value, a base speed of 60 (~24% duty cycle) was chosen and a starting Kp was chosen such that a change from zero error to max error causes a change in the base PWM that is ~50% of the base PWM [9]. Looking at the sensor fusion data from Excel, the difference between the maximum and minimum error was 12000. Since Kp multiplied by 12000 should give 30 PWM, the starting Kp was 0.0025. Since the error is positive when the car is on the right side of the track, the sign of Kp must be positive such that a steering command is sent to the left (we chose the left direction to be positive). To determine the appropriate range of values for Kp, the Kd parameter was set to zero and values of Kp ranging from 0.0025 to 0.04 were chosen. Gradually, Kd was increased until the car no longer ran any successful tests. This allowed each partner to create their own sensitivity tables which included the total lap time for different combinations of Kp and Kd.

Determining appropiate Kp and Kd values on position 1 with a base speed of 60, values provided are time (s)

		Kp value						
Kd value		0.0025 Kp	0.005 Kp	0.01 Kp	0.02 Kp	0.03 Kp	0.035 Kp	0.04 Kp
	0	7.91	0	7.23	7.42	7.1	6.2	0
	0.01	8.3	0	0	8	0	0	9.45
	0.03	0	0	9.3	8.32	0	9.7	10.32
	0.05	8.7	11.4	10.9	7.88	8.22	8.9	8.2
	0.1	7.21	10.1	7.65	0	8.31	8.6	7.86
	0.2	7.2	0	0	8.43	8.38	8.33	7.29
	0.25	7.7	8.09	10.22	8.8	7.67	9.01	8.7
	0.3	0	9.3	0	9.25	7.56	7.9	10.2
	0.35	8.34	7.79	6.89	9.44	7.23	8.2	8.5
	0.4	9.7	6.6	8.5	8.7	6.85	7.7	9.1
	0.5	7.52	7.95	0	6.72	7.5	7.1	0
	0.6	10	0	0	0	0	0	0
	Note: A time of zero indicates that the car failed to complete the track without any errors							

Figure 13a: Kp and Kd sensitivity table for Utkarsh's car

Determining appropiate Kp and Kd values on position 1 with a base speed of 60, values provided are time (s)								
	Kp value							
Kd value		0.0025 Kp	0.005 Kp	0.01 Kp	0.02 Kp	0.03 Kp	0.035 Kp	0.04 Kp
	0	8.08	0.00	7.38	6.58	6.25	5.33	0.00
	0.01	8.47	7.77	0.00	7.17	0.00	0.00	8.65
	0.03	9.21	0.00	9.50	7.49	7.69	8.90	9.54
	0.05	0.00	0.00	10.12	7.05	7.39	8.09	7.37
	0.1	7.36	7.27	7.81	0.00	7.48	7.78	7.03
	0.2	7.35	10.62	9.12	7.61	7.56	7.50	6.44
	0.25	7.86	8.26	8.43	7.98	6.83	8.20	7.88
	0.3	9.90	9.50	0.00	8.44	6.72	7.07	9.41
	0.35	8.52	7.95	7.03	8.64	6.38	7.37	7.68
	0.4	0.00	6.74	8.68	7.88	5.99	6.86	8.29
	0.5	7.68	8.12	0.00	0.00	6.66	6.25	0.00

Figure 13b: Kp and Kd sensitivity table for Avnish's car

The sensitivity tables for both partners are included in Figures 13a and 13b. All the following tests were done at position 1. Furthermore, these initial tests for determining Kp and Kd were done on the straight-line track, not the official curved track required for race day. Furthermore, the (15-14-12-8)/8 weighting schemes were chosen for the Kp and Kd tests, this choice is explained later in this report.

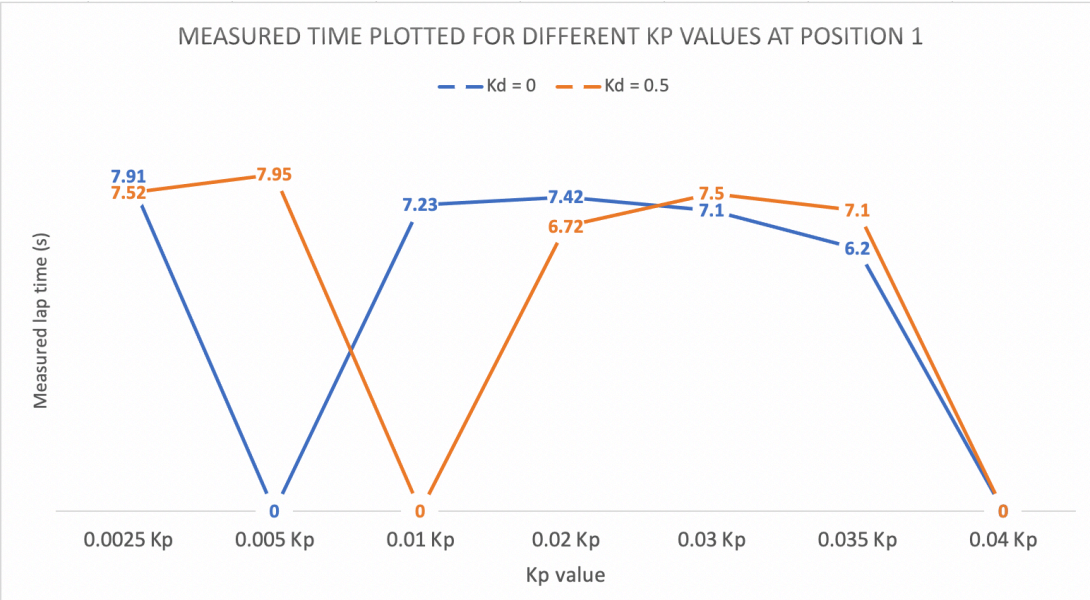


Figure 14: Measured lap time for different values of Kp and at Kd = 0 and Kd = 0.5.

To better visualize which Kp values gave the most consistent and appropriate lap times to the project goals, a line plot was created for the lap times obtained for different Kp values when Kd was 0 and 0.5 specifically. These two Kd values were chosen since they were the two extremes of the data. Figure 14 shows this plot for Utkarsh's data and it can be seen that the 0.02 to 0.035 Kp range gave the lowest time values. A similar analysis was done for Avnish's sensitivity table and the range giving the lowest time values were also 0.02 to 0.04.

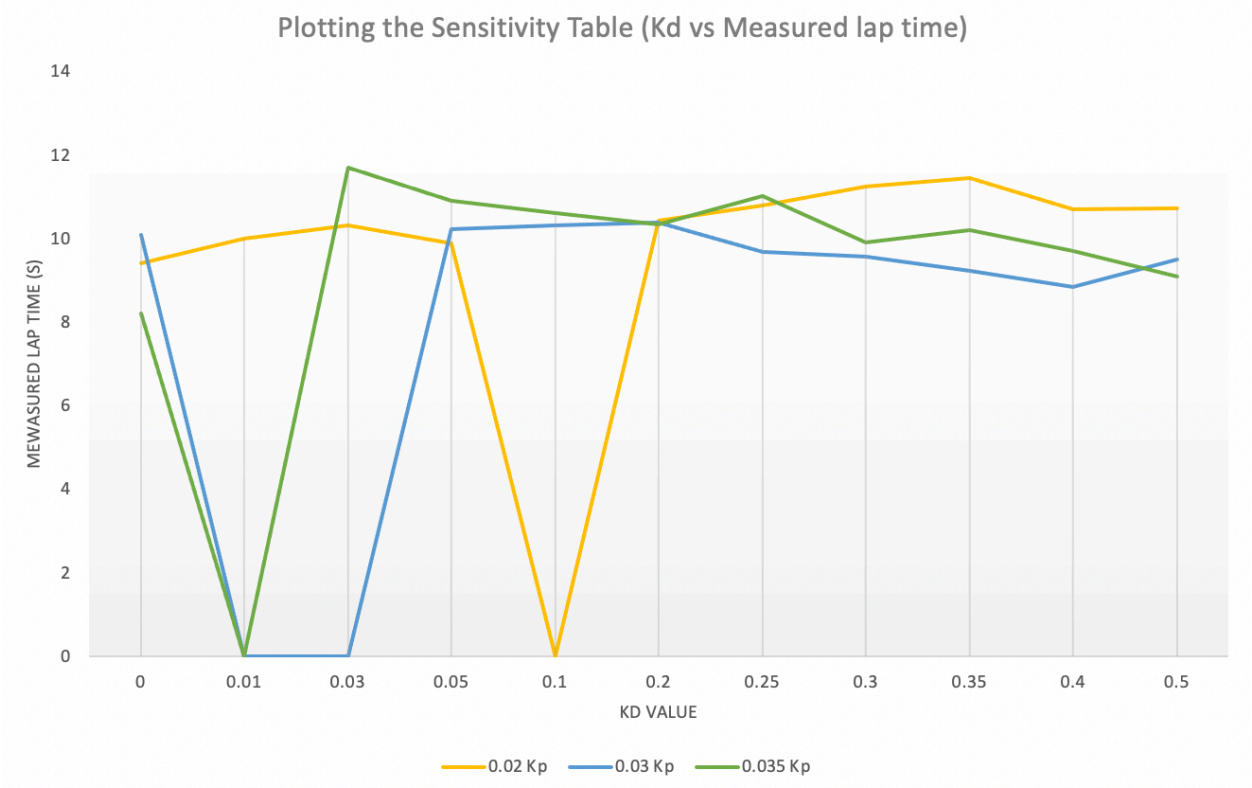


Figure 15: Measured lap time for different Kd values from Position 1 on the straight track (Utkarsh's data displayed here).

Figure 16: A plot shown for the average lap times of Kp and Kd values.

In Figure 16, the plot for Kp was generated by averaging the lap times for each Kp value over the different Kd values. The same calculation was done for Kd against its average lap times for different Kp combinations. The plots above give coefficient of determinations of 0.0008 and 0.0501 for Kp vs. average lap time and Kd vs. average lap time respectively.

Curved Track Runs (Utkarsh)

Base	Kp	Kd	Accel (L/R)	Time	Works for all positions?	Comments
60	0.02	0.4	50	11+	Y	Too slow to achieve goals
60	0.02	0.4	80	10.7	N	Too slow + random errors
70	0.02	0.5	60	10.1	Y	Too close to 11s
70	0.03	0.5	70	9.8	Y	Too close to 11s
70	0.03	0.6	70	0	N	Early donut
70	0.03	0.5	80	9.5	Y	Try higher base speed
80	0.03	0.5	80	8.9-9.0	Y	(Delay of 350)
80	0.04	0.5	80	0	N	Starts going off the track
80	0.03	0.55	80	8.5	N	Runs off
80	0.03	0.6	60	0	N	Early donut
80	0.03	0.5	100	0	N	Runs off
85	0.03	0.5	80	0	N	Runs off
85	0.035	0.5	60	8.8	Y	USED ON RACE DAY
100	0.035	0.5	60	7.8-8.2	Y	Didn't work on race day
100	0.035	0.5	100	0	N	Runs off
100	0.035	0.5	75/90	7.8	N	Didn't work on race day
100	0.04	0.5	80	0	N	Early donut
110	0.035	0.5	100/90	0	N	Random errors
120	0.035	0.5	100/90	0	N	Runs off
120	0.035	0.5	60	0	N	Runs off
120	0.04	0.5	60	0	N	Runs off
Note:	A lap time of zero means the car failed to complete the track					

Figure 17a: Detailed table showing qualitative and quantitative results for different values of the core parameters: base speed, Kp, Kd, and acceleration. The above table is Utkarsh's data.

Curved Track Runs (Avnish)

Base	Kp	Kd	Acceleration	Time	Works for all positions?	Comments
60	0.05	0.6	40	0	N	Runs off track and wobbly
60	0.01	0.55	40	0	N	Runs off track while turning
60	0.025	0.35	40	0	N	Donuts off the track
60	0.02	0.45	40	11.8	Y	Too slow (>11s)
60	0.02	0.45	50	12.1	Y	Too slow (>11s)
60	0.02	0.5	40	0	N	Donuts off early
65	0.02	0.5	60	0	N	Donuts off early
65	0.03	0.45	40	11.25	Y	Too slow (>11s)
65	0.02	0.5	65	0	N	Donuts off early
65	0.03	0.4	40	0	N	
65	0.025	0.45	30	10.2	Y	Too slow
70	0.03	0.45	30	10.8	Y	Too slow for project
70	0.03	0.45	30	**8.4**	Y	USED ON RACE DAY
80	0.02	0.45	30	0	N	Donuts off track
80	0.03	0.45	30	11.4	Y	Too slow (>11s)
80	0.03	0.5	30	0	N	Runs off track
90	0.03	0.45	40	9.7	Y	Unstable on race day
100	0.035	0.45	50	12.3	Y	Too slow (>11s)

Figure 17b: Same table as the one shown in Fig. 17b but for Avnish's data.

100 speed runs				
Positions				
Trial no.	1	2	3	4
1	8.16	7.69	7.71	7.77
2	8.04	7.90	7.70	7.77
3	8.62	8.16	7.97	7.91
	(3 failures)	(3 failures)		
85 speed runs				
Positions				
Trial no.	1	2	3	4
1	8.78	9.01	8.22	8.66
2	8.79	8.42	8.02	8.89
3	8.90	8.35	8.09	8.82
				(1 failure)

Figure 18: Trials conducted by Utkarsh on Race Day at different positions to see the lap time on the curved track with the final chosen parameters (Kp = 0.035, Kd = 0.05) at speeds of 100 PWM and 85 PWM.

Data Interpretation:

Determining a value for Kp and Kd:

Trends spotted in Figures 13a and 14 were that a Kp too high (0.04) or too low (0.005-0.01) had the most failed runs and it was visually observed that for high Kp values, the car appeared to oscillate around the track a lot more and often lose stability leading to most of the failures. Once Kp was narrowed down to the 0.02, 0.03, and 0.035 range, plots were obtained for constant Kp and varying Kd values. This allowed us to find out which Kd value is most appropriate to our project goals. This plot is shown in Figure 15 and there is at least one failed run for each Kp value between the 0-0.1 Kd value range. The results flatten near 0.25 and 0.5 Kd range with 0.03 Kp giving the fastest lap times and 0.02 Kp giving the slowest lap times. This allowed the narrowing down of Kd into a range between 0.2 and 0.5 for the base speed of 60 on the straight line track and a Kp range of 0.02 to 0.035 for Utkarsh's data. Regarding Avnish's data – given in figure 13b – the excel plots indicated a similar range of 0.2 to 0.5 for Kd. It is difficult to determine precisely which combination is best for achieving our project goals since these test runs were conducted on the straight line track hence for the next parameters of interest – base speed and acceleration – we decided to use the curved track. Furthermore, the tests conducted thus far were only done from position 1 hence for the tests conducted on the curved track, all 4 positions were thoroughly tested as shown in the following sub-section.

Some additional trends that were spotted from figures 13, 14, and 15 were as follows:

- When Kd was increased to above 0.5, the car appeared to perform a donut midway through the track (possible reasoning explained in discussion section). This gave several failures for the Kd value of 0.6
- There was no clear direct correlation between Kp and lap time or Kd and lap time, this is shown in Figure 16 showing average lap times when Kd and Kp are held constant respectively and the correlation coefficient values are too low at 0.0008 and 0.05. This meant other parameters, such as base speed, are key to achieving faster times.

From the tests conducted on the straight line track, a reasonable range for appropriate Kp (0.02 to 0.035) and Kd (0.2 to 0.5) values were obtained for both partners. The next step was to increase the base speed from 60 and to work with different values of Kp and Kd with the different values of base speed to see which combinations achieve the project goals of completing the track from all four positions and doing so under 11 seconds (aim was to be averaging below 9.5 seconds allowing for a margin of error).

Finding the right base speed:

This was done by doing various trial runs on the curved track as shown in Figures 17a and 17b, and noting the different errors that would occur for different combinations. The weighting scheme used for these trial runs were also (15-14-12-8)/8. Since several test runs were being conducted, batteries would be replaced every 25 or so trials based on visual observations of the motor's speed and the car's performance on the track. It was observed that the most common indicators of the batteries running out were the car failing to execute the 180 degree turn at the crosspiece and the car's speed significantly reducing causing it to run off as it reaches the curved semi-circular part of the track.

Another parameter of interest here was 'acceleration.' The code we used was designed to increase both the left and right wheel speeds when the car was on the track (i.e. the error value was low enough to approximate the car being at the center). This meant that the car would travel faster when the track ahead is a straight line and would help accomplish the secondary project goal of completing the track under 11 seconds. Different values were tested for this segment of the code, i.e. the speed was increased by different PWM amounts noted in the column labelled 'acceleration' in figures 17a and 17b.

Figures 17a and 17b show a clear pattern between the timings and the base speed, an increase in the base speed reduces the time taken to complete the lap. Another pattern that was noticed between base speeds of 60 and 80 was that Kd needed to be increased to ensure a lap time matching the project goals. This can be seen in Figure 17a between the base speed of 60 and 70 where Kd was increased from 0.4 to 0.5. Kp was also increased as it started at a base value of 0.02 and increased to 0.035 for Utkarsh and 0.03 for Avnish for base speeds of 85 and 70 respectively. However, large increases in Kd for all base speeds resulted in failure for all positions (shown by a 0 second lap time) or a failure in one of the 4 positions (shown by N in the 6^{th} column).

Regarding the final parameter of acceleration, increasing it beyond 80 always resulted in failure and several errors, most notably the car seemed to run off the track when it reached the semicircular arc. Generally it went off-course on the first right turn of the track. Another pattern noticed, not provided in figure 17, was that position 4 from figure 12 posed the greatest problems since the car would run off to the left without detecting the track or it would fail to perform the 180 degree turn on time and keep travelling straight or perform an incomplete turn and travel off course. The discussion section discusses the possible reasons for the error and steps taken to troubleshoot. Finally, the base speed that had the most successful runs appeared to be 70 and 80/85 for both partners. When the base speed was increased in the range above 100, other than one successful run recorded by Utkarsh (shown in 17a), the car appeared to run off and could not stabilize itself in time.

Final trial runs:

Figure 18 shows the lap time data for one of the robotic cars at different base speeds as measured on Race Day. In figure 18, we can see that although the base speed of 100 did not work perfectly, possibly since batteries had to be replaced midway causing the large number of errors in positions 1 and 2, the base speed of 85 worked almost flawlessly despite increasing the average time by approximately a second. This test was done for a base speed of 85, an acceleration of 60, and a Kp and Kd value of 0.035 and 0.5 respectively.

Results

Test Discussion:

The analysis of the data from the previous section was very useful and appropriate to achieving our project goals of getting the car to successfully finish the track in under 11 seconds and from every starting position. The most important observations made were as follows: ideal Kp and Kd ranges are 0.02-0.04 and 0.3-0.5 respectively, a base speed between 70 and 90 is required for the car to finish comfortable under 11 seconds, and the acceleration value at or below the base speed is ideal for stability and completion. The data showed that for Kd values above 0.6, the car struggled to complete the track and performed an early donut. This was likely due to the car lifting up too much while turning causing all of its sensors to read 2500 (since the transmittance in air is 2500). This is the only viable conclusion for why the car performed the early turns. Identifying this was crucial to the project's success and thus Kd was kept below 0.6 for most trials.

Looking at the data in figures 15 and 16, albeit for the straight line track, gave significant insight into how exactly the Kp and Kd values affect the speed at which the car can complete its lap which was directly related to the secondary project goal of getting the car to finish under 11 seconds. The analysis allowed us to narrow down a range for Kp and Kd for which the car can successfully, with minimal failures, complete the track. Figure 16 also showed that since Kp and Kd don't have a direct influence in achieving fast lap times, other parameters must be tested to achieve the goal of 11 seconds. Once the right combination of base speed, Kp, Kd, and acceleration was found after extensive testing, the project goals were successfully achieved since the car for both partners performed consistently from every position.

The consistent results below 11 seconds shown in Figure 18 would not have been possible without the conclusions drawn about Kp and Kd through the data analysis beforehand. When looking at figure 17a and b, base speeds at or above 80 seemed to hardly succeed except for only specific combinations of Kp and Kd. However, the car achieved success with a base speed of 85. Without the data analysis that was done, it would have required hours of trial and error to find a combination of Kp and Kd that worked with a base speed above 80 since even slight variations in Kp and Kd lead to failures. This highlighted the importance of the testing that was done for every single parameter varied within the code since each played a significant role in the effectiveness of the robotic car achieving the project's goals.

Race Day Discussion:

The car performed well on Race Day and successfully achieved the project goals for both partners by completing the track with the starting positions that the TA and LAs asked of. Furthermore, the final track completion time was 8.8s for Utkarsh (highlighted in Figure 17a) and 8.4s for Avnish (highlighted in Figure 17b). Below, each partner highlights their experience with the car on race day as well as obstacles they faced and how they overcame it.

Utkarsh's experience: When testing the car, I expected to have it work with a base speed of 100 on race day, as highlighted in light green in figure 17a, and thus complete the track within 7.8 to 8.2 seconds. On race day, however, after running a few tests there appeared to be lots of problems that arose with a base speed of a 100. Firstly, the car would be very unstable and go off the track constantly. Secondly, upon replacing the batteries, the car would not work for majority of the starting positions, especially starting position 4. Hence, a few last minute changes had to be made. The base speed was decreased to 85 and the batteries were not replaced to avoid any new errors from occurring. Lastly, the weighting scheme had to be modified from (15-14-12-8)/8 to (12-14-12-8)/8 to correct the car from taking an extreme left turn when placed at position 4. Luckily, the car performed successfully from every position, despite taking two attempts from one of the positions to succeed. The goal of the project was thus achieved since it completed the track and did so under 11 seconds.

Avnish's experience: On the morning of Race day, I realized that the original Kp value of 0.04 and Kd values of 0.45 were not working with my car. My car started performing slightly better after replacing it with fresh batteries, but it was still performing the 180 degree turn too early. After running some final track runs, I reduced my Kp to 0.03 to get optimal performance from it. I also tried to improve my time on the track by increasing the base speed from 65 to 70, and was able to lower my race time from 10.2 seconds to 9 seconds on position 2, and that was the configuration I used for Race Day.

The biggest limitation of our code was that it was not universal, i.e. the base speed, Kp, and Kd was specific to the member's track construction and the member's lighting conditions (even after the minimum value sensor array was changed for each member). This meant that if given to a third person or distributed to a different group, their car may not successfully complete the track and achieve the project goals. Another limitation of the code was that it only used a very simplistic model of proportional and derivative controls which didn't make the car very smooth when working at high speeds and led to several errors such as early 180 degree turns and the car randomly running off the track or failing to work for all positions.

There are several ways the project could be conducted differently. The main takeaway from both members was that the more extensive the testing was, the more information was obtained on how to achieve stability at greater speeds. Due to the constraints of time and the limited scope of the project's goals, not enough attempts were made to get the car working at higher speeds and complete the lap at times below 7 seconds. Thus one way to conduct the project differently is to do more testing with Kp and Kd at higher base speeds. Another lesson learnt was that the loop sends speed commands to both wheels in a manner such that they have uneven speed throughout the lap and this contributes to instability. By treating the wheels separately, i.e. having separate Kp and Kd values for each wheel as well as causing the car to turn left or right by sending commands to both wheels, could give greater insight into stabilizing the car. Furthermore, introducing a new parameter and implementing an integral controller may fix the problem of stability and allow the car to work universally from all starting positions.

Conclusion and Future Work

In conclusion, the robotic car successfully completed both the project goals: following the curved track with stability from all four positions and completing each lap under 11 seconds. This was made possible by the sensor fusion process and thorough testing of the core parameters involved in the code: the Kp and Kd value, the base speed, and the acceleration. This project gave us both exposure to how PID controls and logic work and how circuit theory can be applied to path-sensing mechanisms. It gave us both a glimpse into the workings of more advanced control systems such as adaptive cruise control and micro mouses. If we had more time, we would like to have implemented more parameters, such as an integral controller, and conducted similar tests for it to find the ideal combination for stability at high speeds. We believe that although our code and parameters met the project goals, there is a combination of parameters that would give even more consistent results and would allow the car to follow any arbitrary track. Additionally, given more time, we would attempt to make the car follow more complex tracks since it would reveal more information about how Kp and Kd come into play when the car encounters more complex maneuvers.

References

[1] "TIRSLK-EVM." *TIRSLK-EVM Evaluation Board | TI.com*, www.ti.com/tool/TIRSLK-EVM.

[2] "Energia 23." *Energia*, 1.8.10E23, 17 Dec. 2019, energia.nu/download/.

[3] "Graphing Calculator." *Desmos*, www.desmos.com/calculator.

[4] Microsoft Excel 2018 [Computer software].

[5] D. Briggs, *Taking Path Sensor Calibration Measurements.* 2020, https://www.youtube.com/watch?v=swOMZOSCpzM&list=LL&index=12&ab_channel=MikeBriggs

[6] Stafsudd, O.M., EE3 Introduction to Electrical Engineering Laboratory Manual, as updated and appended.

[7] "Pulse Width Modulation Used for Motor Control." *Basic Electronics Tutorials*, 5 Aug. 2020, www.electronics-tutorials.ws/blog/pulse-width-modulation.html.

[8] D. Briggs, "A Conceptual Description of the PID Controller", UCLA, 2020.

[9] D. Briggs, "ECE3 PROJECT TIPS", UCLA, 2020.