ECE3 24F Final Project Report

Nathan Chen (405927069)

Jaycee Alipio (006058711)

Introduction and Planning Our Tests::

The goal of this project was to successfully implement a line-following TI-RSLK (Texas Instrument - Robot Systems Learning Kit) car that could finish the track in under 90 seconds. The TI-RSLK car has eight infrared LED/phototransistor pairs that take inputs every six milliseconds. The phototransistors have variable resistance depending on the light emitted by the infrared LEDs. The higher light intensity the less resistance the phototransistors will have. By using this property of the phototransistor, we can differentiate light and dark conditions by measuring the amount of time needed to discharge a capacitor from fully charged 3.3V to 1V. Light conditions result in shorter measurement times and dark conditions result in longer measurement times. The TI-RSLK car uses these inputs to make sure the car stays centered on the track and make sure that it can detect when it should do a 180 degree turn (at the middle of the track) and when it should come to a stop (at the end of the track).

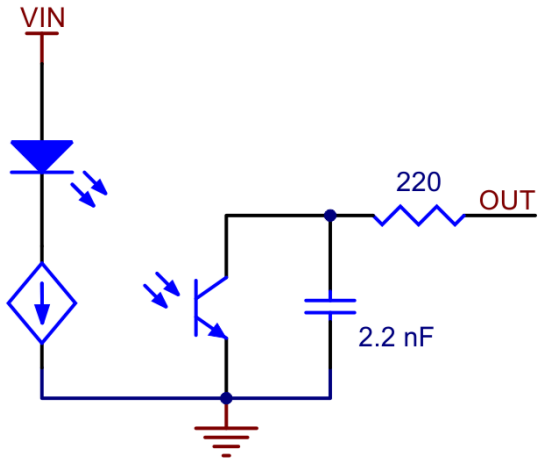


Figure 1. Circuit Diagram of an Individual IR LED/Phototransistor Pair

We utilized this sensor data in a PD controller to dynamically change the motor speeds to stay aligned with the track. The PD controller consists of two parts, the proportional and derivative controllers. The proportional controller is responsible for giving a speed change command to the wheels based on a proportion of the error value. The proportional controller is good at adjusting the current position to the line on the track, but it may overshoot because it is ignorant of the velocity of the car. The derivative controller compensates for the proportional controller by looking at the difference of the current error and the previous error. The derivative controller helps the car's oscillation by decreasing the speed of the motors when the velocity of the car is high. Additionally we tested the car with encoder counts for some parts of the project, as it would help us to independently adjust the speed. The wheels of the car were powered with PWM (pulse width modulation). The car sends pulses of a square wave signal at extremely high frequencies to simulate a constant voltage. This helps us vary the wheel speed from 0 to 255 by modifying the duty cycle of the car.

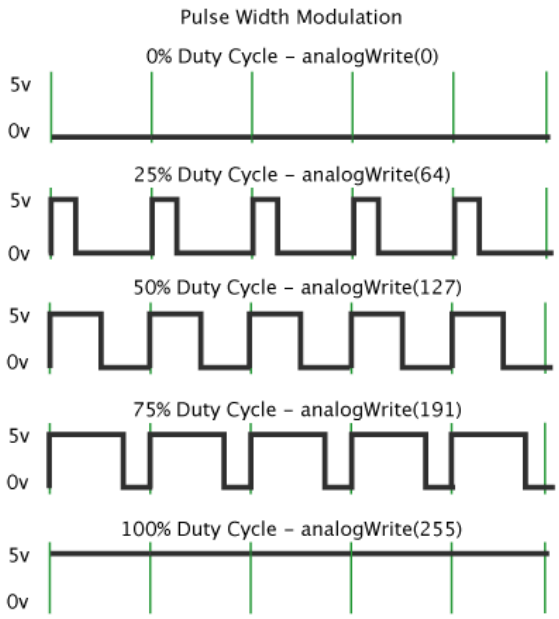


Figure 2. Diagram of PWM

Our independent variables in these experiments are speed, Kp, Kd, start position, and the battery voltage. The dependent variables of this experiment are the oscillation rating (a way we quantified the quality of how well the TI-RSLK car was following the track), and the marker where the car left the track. The variables that we kept constant were the lighting conditions and the hardware on the robot.

The arduino code that we modified during the experiments are also important in planning our tests. Every time we changed the code, we kept a log of what we changed and how it affected it in the next test. Additionally, we collected all of our raw data in a log book and used prior results to plan out our next test. For example, if the car struggled to stay in a straight line, we tried lowering the Kp or increasing the Kd value.

Conducting Our Tests:

The first thing we did once we entered the lab was to check the battery voltage of the car. We found that once the battery voltage reaches around 6 V, the motor response becomes very sluggish and the car will have difficulty moving. Every 20 data points we would re-measure our battery voltage in our log. Throughout the whole project, we used the following weights for the eight sensors: {-15, -14, -12, -8, 8, 12, 14, 15}. We made adjustments to the Kp and Kd values based on the previous trials with the goal of reducing oscillations and increasing line following capabilities.

We conducted each test by placing the car at the starting position on the track. We used starting position 2 for most of the tests to make it easier to debug code and adjust the Kp and Kd values. Once we did get successful runs, we changed the starting position to make sure that the car can successfully do the track on different starting positions. We recorded the key results such as the marker where the car leaves the track and near the second half of our project we started recording the oscillation rating. Our oscillation rating is defined as 1 being the worst and most jittery to 5 as the best and most stable oscillations. We also

recorded other key behavior such as doing the 180 degree turn too early. Lastly we included any changes we made to the arduino code in our log book.

Analyzing Our Data:

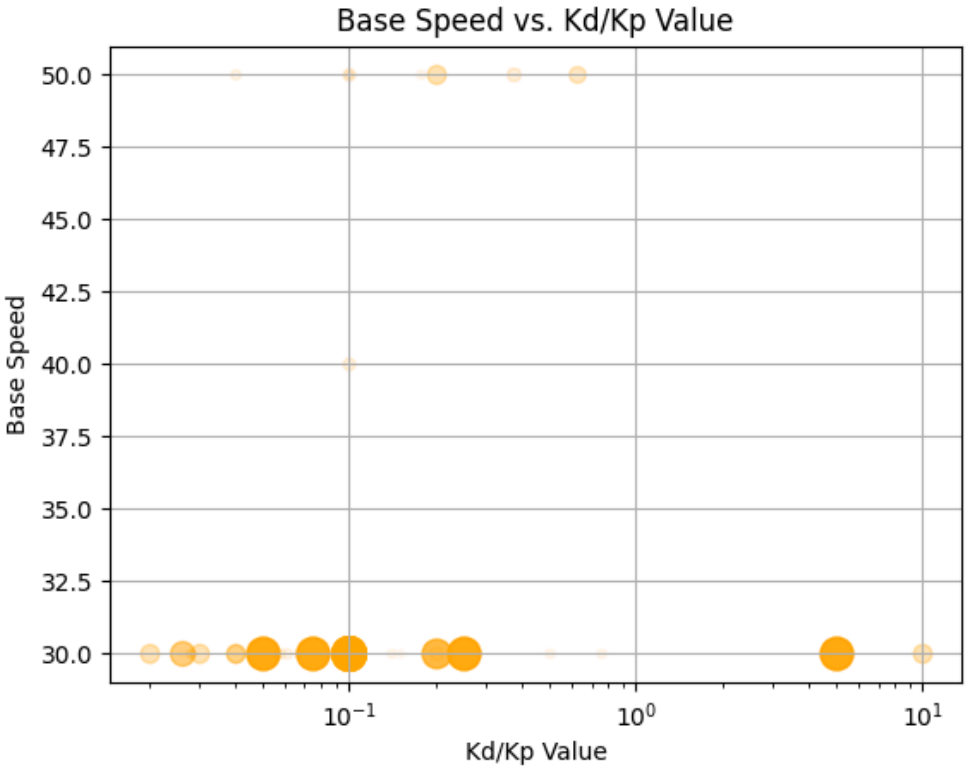


Figure 3. Distribution of Finish Position (marker number) as a function of Kd/Kp ratio and Base Speed

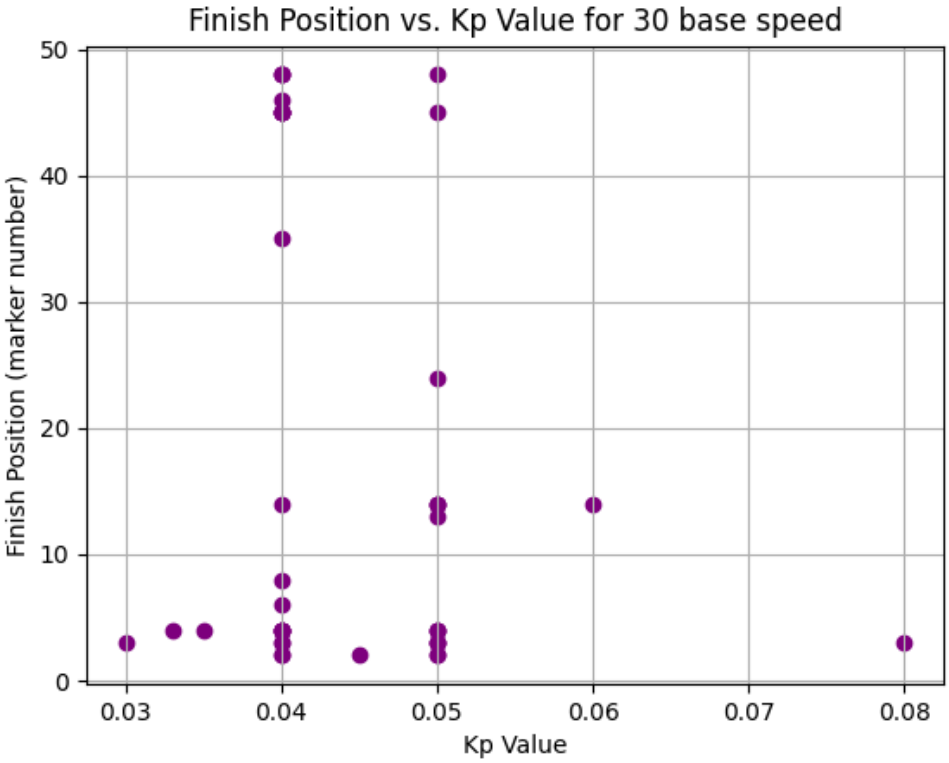


Figure 4. Relationship between Finish Position (marker number) and Kp value.

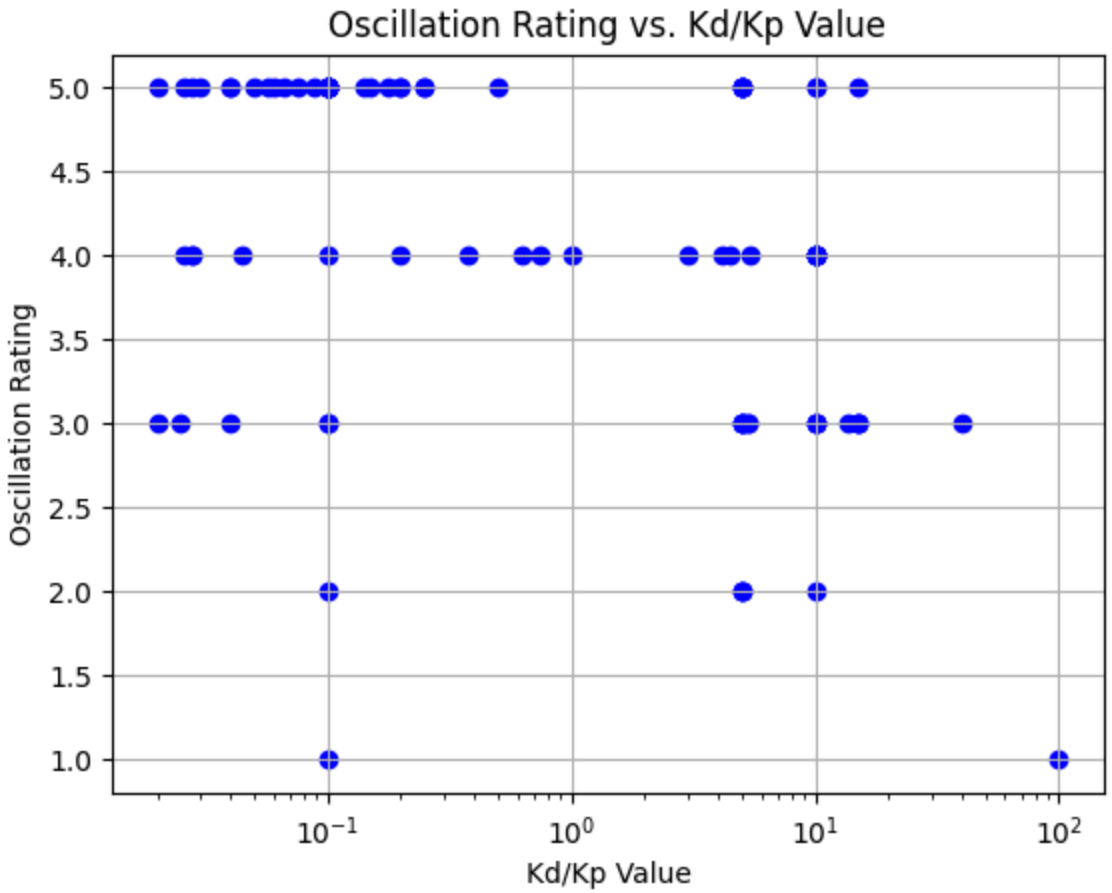


Figure 5. Relationship between Kd/Kp value and Oscillation Score (1 is worst and 5 is best).

Interpreting Our Data:

Our Arduino code had four major changes during the whole project. The first iteration of our code, we made it so that the Kp and Kd would iteratively add speed onto the motors, which resulted in the motors reaching max speed really quickly. This resulted in the car speeding off the track even though our base speed was set to a low value, 20. We fixed this by adding a local variable to calculate the speed command that was sent to the motors instead of directly using the base speed.

Our second version of the code simply included the PD controller. Initially we were using Kp and Kd values that were far too high. As a result, the car would oscillate with high overshoot and would skip portions of the track, such as the loop at marker 03. Our initial Kp value was derived by using this formula: (Error Margin) (Base Speed) = Kp (Max Error Value), where error margin was between 50% and 100%. The Max Error Value was taken from our calibrated sensor fusion plot. Additionally we used Kd values that were about 5 to 10 times the ratio of Kp. Eventually we found a set of values that had a very good oscillation rating (Base speed = 30, Kp = 0.0075, Kd = 0.0525, Oscillation Score = 5). Still this version of the arduino code had trouble following the sharp turn at marker 03. The car would always turn towards the loop, but would eventually leave the track near the middle of the loop.

To progress through the loop, we used encoder counts in our third iteration of the arduino code. The difficult sharp turns were handled independently of the PD controller. We left the base speed at 30,

Kp at 0.0075, and Kd at 0.00525 since the car was following the line very well on the straighter sections of the track. The starting tick for the encoder turn and the encoder tick duration was found for every sharp turn through trial and error. Using encoder counts to hard code the sharp turns on the track, we completed the track a couple of times using starting position 2. However, when we changed the starting position of the car, the car would end up turning early or late at certain parts of the track. We decided to change our code because using encoder counts was far too inconsistent and required many runs before the car could complete the track.

Our last iteration of the code was a modified version of the second iteration of the arduino code. We added a type of movement command when the PD value was too high. When the PD value was greater than twice the base speed, we reversed one of the wheels (if PD was positive → reversed left wheel, if PD was negative → reversed right wheel). Initially we had weights of 0.5 on each wheel, but we eventually settled on 0.25 on each wheel to help the car make sharper turns. We observed that this helps the car not understeer the turns and lets the sensors stay on top of the line. Lastly we had a problem with the car detecting crosspieces. Sometimes the car has more instances where all sensors would read 2500 than the expected number of times on the track. This made the car do a 180 degree turn at random spots. To solve this problem, we used what we learned from encoder counts to have an encoder range for when the car would start detecting crosspieces. This considerably increased the amount of successful runs we had on the track.

For simplicity, we have only graphed the Base Speed vs. Kp/Kd with varying opacity of how far the car went on the track for the last iteration of our code. A run that went all the way to the end of the track (marker 48) was given 100% opacity and scaled to a bigger area, while a run that went only to marker 24 was given 50% opacity. We decided not to graph the first and second iterations because the graph was consistently leaving the track around the first 4 markers of the track. This would mean for the same Kp/Kd values that we used for the successful runs, it would have much higher discrepancies between the level of success. For example, for Kp = 0.04 and Kd = 0.004, the second iteration consistently went to marker 03 while the last iteration of the code went through the whole track. From graphing the data, we found that the range of Kd/Kp values that worked best for completing the track was around 0.05 to 0.2. Kd was around 20 to 5 times smaller than Kp. This is a very generous range of values. Additionally there was another case where the Kd was 5 times bigger than the Kp value and the run went until marker 45. The Kd value seems to not matter as much for staying on the track. The Kp value was more important in determining whether the car finished the track. A Kp value of 0.4 or 0.5 for 30 base speed had more successful runs than other Kp values. However there were a couple of runs in for these Kp values that did not do as well on the track. These runs could be attributed to inconsistently detecting the cross pieces and doing a 180 degree turn too early or having high value weights on the wheels for sharp turns. When we decreased the wheel speed weights on sharp turns, we began to see more consistent results on the turns.

Generally we found that a Kd/Kp ratio of 0.05 to 0.15 gave a pretty good oscillation score. As the Kd/Kp ratio increased above the range, the oscillation score began to decrease. A lower Kd/Kp ratio implies that the proportional gain dominates the speed command. With higher Kp, it ensures that the TI-RSLK car can quickly correct and have smoother oscillations. With a higher Kd/Kp ratio the Kd ends up dominating the speed command to the motors. This results in a more overdamped trajectory towards the line and a bigger oscillation on the first set of input commands.

On race-day the car completely finished the track with 86 seconds!

Appendix:

"Pololu - 8-Channel QTRX Sensor Array for ROMI/TI-RSLK Max (Headers Not Soldered)."

Pololu Robotics & Electronics, www.pololu.com/product/3545. Accessed 3 Dec. 2024.

Hirzel, Timothy. *Basics of PWM (Pulse Width Modulation)*, 15 Dec. 2022,

docs.arduino.cc/learn/microcontrollers/analog-output/. Accessed 3 Dec. 2024.

Calibration Data:

Calibration Data directly from IR Sensors

Sensor Fusion Graph derived from Calibration Data

Arduino Code:

```
#include <ECE3.h>

uint16_t sensor_values[8];


const float min_array[] = {919, 708, 709, 709, 686, 640, 731, 732};
const float max_array[] = {1581, 1792, 1791, 1791, 1814, 1860, 1769, 1768};
const float weights[] = {-15, -14, -12, -8, 8, 12, 14, 15};


const int left_nslp_pin = 31;       // not sleep
const int left_dir_pin  = 29;       // rotation directions
const int left_pwm_pin  = 40;       // pulse width modulation (controls motor speed)

const int right_nslp_pin=11;
const int right_dir_pin=30;
const int right_pwm_pin=39;

const float Kp = 0.04;
const float Ki = 0;
const float Kd = 0.004;
```

```
const int base_speed = 30;              // (error margin)(Base speed) = Kp (maxErrorValue)

int current_right_speed;
int current_left_speed;

int sensor_max_reading;
bool has_turned = false;

float p_value;
float i_value = 0;
float d_value;
float error;
float previous_error = 0;
float pid_value;


void setup(){
  pinMode(left_nslp_pin,OUTPUT);
  pinMode(left_dir_pin,OUTPUT);
  pinMode(left_pwm_pin,OUTPUT);

  pinMode(right_nslp_pin,OUTPUT);
  pinMode(right_dir_pin,OUTPUT);
  pinMode(right_pwm_pin,OUTPUT);

  digitalWrite(left_nslp_pin,HIGH);
  digitalWrite(right_nslp_pin,HIGH);

  digitalWrite(left_dir_pin,LOW);
  digitalWrite(right_dir_pin,LOW);

  ECE3_Init();
  Serial.begin(9600);
  resetEncoderCount_left();
  resetEncoderCount_right();
   delay(2000);
}


int average_encoder_count() {
  return ((getEncoderCount_left() + getEncoderCount_right())/2);
}


void loop(){
  error = 0;
  sensor_max_reading = 0;
  ECE3_read_IR(sensor_values);
  for(int i = 0; i < 8; i++) {
    if(sensor_values[i] == 2500) {
      sensor_max_reading++;
    }
```

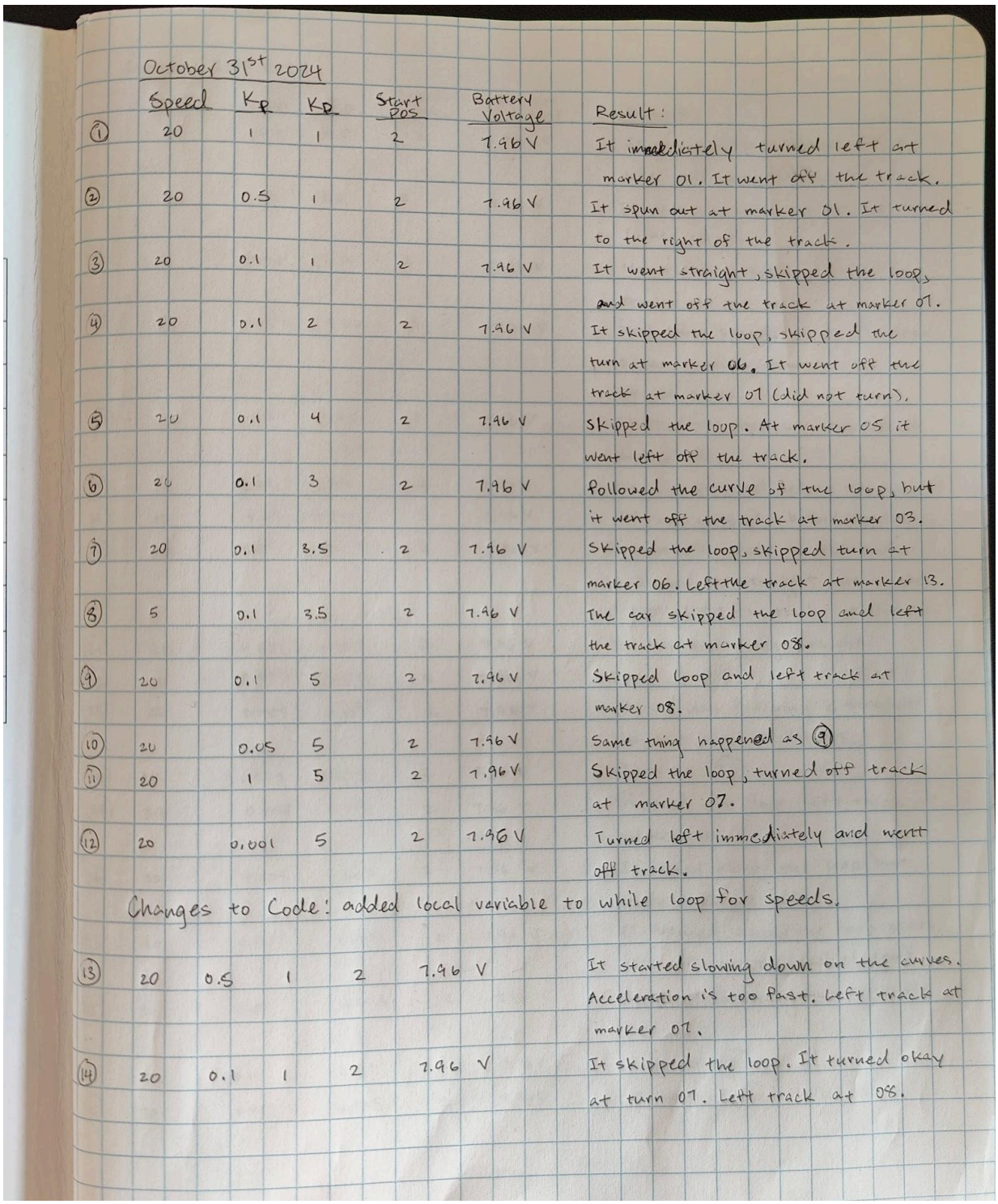
```
    sensor_values[i] -= min_array[i];
    sensor_values[i] = sensor_values[i] * 1000 / max_array[i];
    error += sensor_values[i] * weights[i];
  }
  if (average_encoder_count() > 6500 && average_encoder_count() < 7000 && !has_turned) {
    if (sensor_max_reading == 8) {
      digitalWrite(left_dir_pin,LOW);
      digitalWrite(right_dir_pin,HIGH);
      analogWrite(right_pwm_pin, 50);
      analogWrite(left_pwm_pin, 50);
      delay(1500);
      digitalWrite(left_dir_pin,LOW);
      digitalWrite(right_dir_pin,LOW);
      has_turned = true;
    }

  }
  if (average_encoder_count() > 13500 && average_encoder_count() < 15000) {
    if (sensor_max_reading == 8) {
      digitalWrite(left_nslp_pin,LOW);
      digitalWrite(right_nslp_pin,LOW);
    }
  }

  error /= 8.0;
  p_value = error * Kp;
  i_value += error * Ki;
  d_value = (error - previous_error) * Kd;
  pid_value = p_value + i_value + d_value;
  if (pid_value > base_speed * 2) {   //sharp left
    int saved = average_encoder_count();
    while (average_encoder_count() - saved < 4) {
      digitalWrite(left_dir_pin, HIGH);
      analogWrite(right_pwm_pin, 0.25 * (base_speed + pid_value));
      analogWrite(left_pwm_pin, 0.25 * (base_speed + pid_value));
    }
    digitalWrite(left_dir_pin,LOW);
  } else if (pid_value < -1 * base_speed * 2) { // sharp right
    int saved = average_encoder_count();
    while (average_encoder_count() - saved < 4) {
      digitalWrite(right_dir_pin, HIGH);
      analogWrite(right_pwm_pin, 0.25 * (base_speed - pid_value));
      analogWrite(left_pwm_pin, 0.25 * (base_speed - pid_value));
    }
    digitalWrite(right_dir_pin, LOW);
  } else {    // straight line
    analogWrite(right_pwm_pin, base_speed + pid_value);
    analogWrite(left_pwm_pin, base_speed - pid_value);
  }
  previous_error = error;

}
```

Raw Data from Log Book:



October 31st 2024

	Speed	K_p	K_D	Start Pos	Battery Voltage	Result:
(1)	20	1	1	2	7.96 V	It immediately turned left at marker 01. It went off the track.
(2)	20	0.5	1	2	7.96 V	It spun out at marker 01. It turned to the right of the track.
(3)	20	0.1	1	2	7.96 V	It went straight, skipped the loop, and went off the track at marker 07.
(4)	20	0.1	2	2	7.96 V	It skipped the loop, skipped the turn at marker 06. It went off the track at marker 07 (did not turn).
(5)	20	0.1	4	2	7.96 V	Skipped the loop. At marker 05 it went left off the track.
(6)	20	0.1	3	2	7.96 V	Followed the curve of the loop, but it went off the track at marker 03.
(7)	20	0.1	3.5	2	7.96 V	Skipped the loop, skipped turn at marker 06. Left the track at marker 13.
(8)	5	0.1	3.5	2	7.96 V	The car skipped the loop and left the track at marker 08.
(9)	20	0.1	5	2	7.96 V	Skipped loop and left track at marker 08.
(10)	20	0.05	5	2	7.96 V	Same thing happened as (9)
(11)	20	1	5	2	7.96 V	Skipped the loop, turned off track at marker 07.
(12)	20	0.001	5	2	7.96 V	Turned left immediately and went off track.

Changes to Code: added local variable to while loop for speeds.

(13)	20	0.5	1	2	7.96 V	It started slowing down on the curves. Acceleration is too fast. Left track at marker 07.
(14)	20	0.1	1	2	7.96 V	It skipped the loop. It turned okay at turn 07. Left track at 08.

October 31st 2024

	Speed	Kp	Kd	Start Pos	Battery Voltage	Result:
(15)	20	0.1	3.0	2	7.96 V	It turned along the loop, left half way through the loop
(16)	5	0.1	3.0	2	7.96 V	It went off track right away. Turned right.
(17)	5	0.01	3.0	2	7.96 V	Much faster speed, went off track at the loop.
(18)	5	0.01	0.01	2	7.96 V	Much lower speed, the car stopped after turning towards the line at the start.
(17)	5	0.01	0.02	2	7.96 V	Skipped the loop, car stopped at marker 06.
(18)	5	0.05	0.02	2	7.96 V	The car stopped before the loop
(19)	10	0.05	0.02	2	7.96 V	The car stopped before the loop
(20)	20	0.05	0.02	2	7.96 V	It went off track to the end. It followed the straight until the U-turn. Left the track at U-turn
(21)	20	0.001	0.001	2	7.96 V	The car kept going straight.
(22)	20	0.002	0.001	2	7.96 V	Same as (21)
(23)	20	0.003	0.001	2	7.96 V	The car was following a straight line, but with oscillations.
(24)	20	0.003	0.002	2	7.96 V	Same as (23)
(25)	30	0.003	0.002	2	7.96 V	Skipped the loop, left track at marker 06.
(26)	30	0.005	0.002	2	7.96 V	It turned along the loop but went straight off.
(27)	30	0.01	0.002	2	7.96 V	It followed around the loop better, flew off near the end of loop
(28)	30	0.01	0.005	2	7.96 V	Oscillated on straights, left off the loop.
(29)	30	0.01	0.01	2	7.96V	Same as (28)
(30)	30	0.01	0.015	2	7.96V	It went off track before loop

November 7th 2024

Note: Added threshold for when |error| > 2000, will turn off one motor and make the other run at base speed.

	Speed	K_p	K_p	Start Pos	Battery Voltage	Results:
①	30	0.0075	0.0375	2	7.03 V	Went off the track at milestone 3. ↳ Best run so far.
②	50	0.0075	0.045	2	7.03 V	Went off the track at milestone 3.

Added a delay (2000) when the threshold for |error| > 2000.

③	30	0.0075	0.045	2	7.03 V	Went off track at milestone 3.

changed delay to 6 seconds.

④	30	0.0075	~~0.045~~ 0.045	2	7.03 V	Went off track at milestone 3.

Do_turn condition not applied to error < -2000, fixed.

⑤	50	0.0075	0.045	2	7.03 V	Did a 180 turn at the start.

Got rid of threshold conditionals, implemented encoder counts to know where loops are. Range from 1500-1600 encoder counts, speed=50

⑥	30	0.0075	0.045	2	7.03 V	Went off track at milestone 3.

Changed the range to 1500-2500 encoder counts.

⑦	30	0.0075	0.045	2	7.03 V	Went off track at milestone 3. The encoder case turned on way after loop.

Changed range to 500-1500 encoder counts.

⑧	30	0.0075	0.045	2	7.03 V	It completed the loop at milestone 3 but it did an extra $\frac{1}{2}$ a loop. Went off on milestone 3.

Changed range to 500-1000 encoder counts.

⑨	30	0.0075	0.045	2	7.03 V	Went off the track at milestone 6.

Nov 7, 2024

	Speed	Kp	Kd	Start Pos	Battery Voltage	Results:
(12)	30	0.0075	0.045	2	7.84 V	Added encoder range (1500, 1750) Spun left at marker 05.
		Changed encoder range to (2000, 2250)				
(13)	30	0.0075	0.045	2	7.84 V	Left track at marker 06.
		~~Changed~~ ecoder Range to (1750, 2000)				
(14)	30	0.0075	0.045	2	7.84 V	Went off at ~~05~~ marker 07
		Changed encoder range to (1700, 1950).				
(15)	30	0.0075	0.045	2	7.84 V	Turned too far to left. Left at marker 07.
		changed encoder range to (1650, 1800)				
(16)	30	0.0075	0.045	2	7.84 V	Cut the corner at marker 08. Left the track at 13.
		changed encoder range to (1650, 1850) Added encoder range for (3500, 3700)				
(17)	30	0.0075	0.045	2	7.84 V	Turned at the u-turn, left at marker 14
		changed encoder range (3450, 3750)				
(18)	30	0.0075	0.045	2	7.84 V	Left track at marker 14.
		changed encoder ~~(3450~~ 3400, 3800)				
(19)	30	0.0075	0.045	2	7.84 V	It jumped over the line at marker 8.
		changed 2nd encoder range (1650, 1800)				
(20)	30	0.0075	0.045	2	7.84 V	Skipped the corner at marker 08. Left the track at 14.
		changed encoder count ranges (1650, 1825), (3400, 3850).				
(21)	30	0.0075	0.045	2	7.84 V	Left the track at marker 15.
		change encoder range (3400, 3825				
(22)	30	0.0075	0.045	2	7.84 V	The car left at marker 15.
		added a change direction command, speed = 25.				
(23)	30	0.0075	0.045	2	7.84 V	It spun in a circle at the u-turn.

Added functions for do_turn and do_u_turn. Both take in left and right speed, as well as number of encoder counts to turn for. do_turn has both wheels facing forward while do_u_turn has right wheel reversed and left wheel forward.

Nov 14th 2024

Added functions to do turn and do u-turn. This takes the # number of encoder ticks instead of a range.

	Speed	Kp	Kd	Start Pos	Battery Voltage	Results:
(1)	30	0.0075	0.045	2	7.98 V	It did the u-turn for too long, left track at marker 14.
		changed right speed to 20, left speed to 30. # of encoder ticks = 400.				
(2)	30	0.0075	0.045	2	7.98 V	Left track at marker 14 again.
(3)	30	0.0075	0.045	2	7.98 V	Left track at marker 24.
		Added a do_turn call at 5000 encoder ticks for 50 ticks				
(4)	30	0.0075	0.045	2	7.98 V	It fails at the loop in the beginning. The wheels started moving in pulses.
		changed # ticks of ~~middle~~ second turn to 200.				
(5)	30	0.0075	0.045	2	7.98 V	The car made it to half way.
		changed second turn start to 1650 ticks, u-turn start to 3500 ticks				
		added u-turn at 5850 ticks. for 350 ticks				
		added turn at 5400 ticks for 50 ticks				
		added u-turn at 6950 ticks for 350 ticks.				
(6)	30	0.0075	0.045	2	7.98 V	did a u-turn at marker 9.
		commented out the new turns.				
(7)	30	0.0075	0.045	2	7.98 V	made it to the ~~end~~ middle of the track, marker 25.
		changed the turn before the end to 75 ticks.				5 ticks. ↓
		uncommented the u-turn at marker 25. added a gap between function calls				
(8)	30	0.0075	0.045	2	7.98 V	did not do a u-turn at 13.
		Reimplemented ~~codes~~ code using encoder ranges.				
(9)	30	0.0075	0.045	2	7.98 V	left track at marker 8.
		Reimplemented code without ranges.				
(10)	30	0.0075	0.045	2	7.98 V	did a u-turn at 19.
		removed the turn at 5000 ticks.				

Nov. 14 2024

	Speed	Kp	Kp	Start Pos	Battery Voltage	Results
(11)	30	0.0075	0.045	2	7.98 V	Left track at marker 24. Went straight.
	added turn at 6000 encoder ticks.					
(12)	30	0.0075	0.045	2	7.98 V	turned left off track at marker 23.
	changed turn from 6000 → 6200 ticks					
(13)	30	0.0075	0.045	2	7.98 V	made it to marker 25.
	added u-turn at 6300 ticks.					
(14)	30	0.0075	0.045	2	7.98 V	made it to marker 35. Did the
	changed u-turn from 6300.					u-turn at marker 25 too early.
	to 6350 ticks. added turn at 6800 ticks.					
(15)	30	0.0075	0.045	2	7.98 V	made it to marker 35.
	added u-turn at 9500 ticks for 350 ticks.					
(16)	30	0.0075	0.045	2	7.98 V	the u-turn was way too late.
	changed u-turn from 9500 → 9350 ticks.					
(17)	30	0.0075	0.045	2	7.98 V	the u-turn was a little bit late,
	changed turns from 6350 → 6400 ticks					
	6800 → 6900 ticks.					
(18)	30	0.0075	0.045	2	7.98 V	Left the track at marker 42.
	added turn at 11350 encoder ticks. for 200 ticks.					
(19)	30	0.0075	0.045	2	7.98 V	Left at marker 45.
	added turn at 12000 encoder ticks for 500 ticks					
	added stop at 17000 encoder ticks.					
(20)	30	0.0075	0.045	2	7.98 V	left the track on marker 25
	added a delay on the turn at marker 25. by 50 ticks before and after.					
	added delay to all turns after this by 100 ticks					
(21)	30	0.0075	0.045	2	7.98 V	left the track at marker 45.

Nov 17th 2024

	Speed	Kp	Kd	Start Pos	Battery Voltage	Results:
①	30	0.0075	0.045	2	7.839 V	Went off track at marker 24.
	increased turn at 6200 ticks from 50 ticks to 75 ticks.					
②	30	0.0075	0.045	2	7.839 V	left track at marker 45.
③	30	0.0075	0.045	2	7.839 V	Left track at marker 42.
	↑changed u-turn at marker 24 from 6450 to 6500.					
	decrease the uturn at marker 24 from 6500 to 6400.					
	changed turn ~~3~~ from 7000 ticks to 6850 ticks					
④	30	0.0075	0.045	2	7.839 V	did the turn at marker 43 too late. ~~early~~
	changed start of turn from 11400 to 11300,					
	start of turn from 13000 to 12700 ~~00~~					
⑤	30	0.0075	0.045	2	7.839 V	left the track at marker 24.
	increased u-turn duration from 350 to 375 ticks. changed all turn speeds to 30.					
⑥	30	0.0075	0.045	2	7.839 V	Left the track at marker 44.
	changed last turn from 12700 tick to 12000 starting tick.					
⑦	30	0.0075	0.045	2	7.839 V	Left the track at marker 46.
	changed last turn duration 500→600 ticks.					
⑧	30	0.0075	0.045	2	7.839 V	The car finished the track!
						the last loop was slightly off
	delayed the start of last turn from 12000 to 12100 ~~00~~					
⑨	30	0.0075	0.045	2	7.839 V	The ~~turn~~ last turn was missed, left track
						at marker 45.
	change start of last turn from 12100 ~~00~~ to 12050 ticks.					
⑩	30	0.0075	0.0525	2	7.839 V	the car started the loop a bit later.
	changed start of last turn from 12050 to 12000 ticks.					
⑪	30	0.0075	0.0525	2	7.839 V	Left track at marker 46. Early turn ~~Late~~
	changed the start of last turn from 12050 to 12100 ticks.					
⑫	30	0.0075	0.0525	2	7.839 V	started last turn too late
	changed the start of last turn from 12100 ~~to~~ 12075.					
⑬	30	0.0075	0.0525	2	7.839 V	Late for turn again.
⑭	30	0.0075	0.0525	3	7.839 V	went off track at 046.
	changed last turn start tick to 12025.					
	changed turn speed at marker 08 to 20.					

Nov 17th 2024

	Speed	Kp	Kd	Start pos	Batt. Volt.	Results!
15	30	0.0075	0.0525	2	7.839V	Finished track. the last turn is off from the middle.
16	30	0.0075	0.0525	3	7.839V	left track at marker 46.

changed the turn at marker 08 from 1650 tick to 1600 ticks.
added turn at 17000 tick for 100 ticks.

17	30	0.0075	0.0525	2	7.839V	left track at marker 08.

changed the turn at marker 08 to a u-turn. (10 left speed 30 right speed)

18	30	0.0075	0.0525	2	7.839V	left track at marker 08.

changed left speed of u-turn to 15.

19	30	0.0075	0.0525	2	7.839V	left turn at marker 08.

changed turn speed from 15 → 20.

20	30	0.0075	0.0525	2	7.839V	did marker 15 too early

added a delay of 100 ticks for each turn.

21	30	0.0075	0.0525	2	7.839V	u-turn at marker 42 too late.

Delayed the u-turn at marker 42 by 100 ticks
delayed all functions after by 100 ticks.

22	30	0.0075	0.0525	2	7.839V	the turn at marker 42 was late.

change start of turn from 11500 to 11400.

23	30	0.0075	0.0525	2	7.839V	left at marker 44

changed turn at marker 43 from 17300 → 17350 tick start pos
from 100 tick → 150 tick duration.

24	30	0.0075	0.0525	2	7.839V	left at marker 44 again.

changed duration from 150 to 200 tick duration.

25	30	0.0075	0.0525	2	7.839V	left track at 44 again, skipped loop.

changed duration of turn from 200 → 300 ticks.

26	30	0.0075	0.0525	2	7.839V	left track at 44.

changed ~~duration of turn from 300 → 400 ticks~~
start pos of turn to 12250 and last turn to 12000 ticks.

27	30	0.0075	0.0525	2	7.839V	left the track at 44

changed duration from 300 → 350.

28	30	0.0075	0.0525	2	7.839V	left track at 44 again.

got rid of last turn. added turn at 12400 for 500 ticks.

29	30	0.0075	0.0525	2	7.839V	started turn too late

Nov 17th 2024

changed 12400 → 12100 start pos of turn.

	Speed	Kp	Kd	start Pos	Batt. Volt.	Results:
(30)	30	0.0075	0.0525	2	7.839 V	turned at the final loop too early.

changed 12100 → 12200 start pos

(31)	30	0.0075	0.0525	2	7.839 V	running to the right where the start of the track is. left at 44.

~~changed~~ added turn at 12000 for 100 ticks

(32)	30	0.0075	0.0525	2	7.839 V	leaving track at marker 44.

added a portion of code to go straight at 12100 ticks.

(33)	30	0.0075	0.0525	2	7.839 V	left track at marker 46.

Decided to comment out encoder related code, it is inconsistent when using different starting position.

Nov 21 2024

(1 is worst, 5 is best)

	Speed	Kp	Kd	Start Pos	Battery Voltage	Oscillation Rating	Results:
(1)	50	0.0125	0	2	9.43 V	3	Left at marker 02, oscillated slowly over the line.
(2)	50	0.01375	0	2	9.43 V	2	Left at marker 03, oscillated more that previous trial.
(3)	50	0.015	0	2	9.43 V	2	Left at marker 03
(4)	50	0.01625	0	2	9.43 V	2	Left at marker 03
(5)	50	0.0175	0	2	9.43V	2	Left at marker 03
(6)	50	0.01875	0	2	9.43 V	2	Left at marker 03, much more sneppy
(7)	50	0.02	0	2	9.43 V	1	Left at marker 02
(8)	50	0.02125	0	2	9.43 V	1	Left at marker 02.
(9)	50	0.02125	0.10025	2	9.43 V	3	Left at marker 03.
(10)	50	0.0225	0.1125	2	9.43 V	3	Left at marker 03.
(11)	50	0.02375	0.11875	2	9.43 V	3	Left at marker 03
(12)	50	0.025	0.125	2	9.43 V	3	Left at marker 03
(13)	50	0.02625	0.13125	2	9.43 V	3	Left at marker 03
(14)	50	0.03	0.15	2	9.43 V	2	Left at marker 03
(15)	50	0.035	0.175	2	9.43 V	2	Left at marker 03
(16)	50	0.04	0.2	2	9.43 V	2	Left at marker 03
(17)	50	0.05	0.5	2	9.43 V	4	Left at marker 03
(18)	50	0.07	0.7	2	9.43 V	4	Left at marker 03
(19)	50	0.09	0.9	2	9.43 V	~~4~~ 3	Left at marker 02, did not react to the turn
(20)	50	0.08	0.8	2	9.35 V	3	Same as before, left at marker 02
(21)	50	0.075	0.75	2	9.35 V	3	Left at marker 03.
(22)	50	0.076	0.76	2	9.35 V	3	Left at marker 03.
(23)	50	0.0755	0.755	2	9.35 V	4	Left at marker 03.
(24)	50	0.08	1.2	2	9.35 V	3	Left at marker 02.
(25)	50	0.075	1.125	2	9.35 V	3	Left at marker 02
(26)	50	0.078	1.17	2	9.35 V	3	Left at marker 02.
(27)	30	0.045	0.45	2	9.35 V	4	Left at marker 02.
(28)	30	0.04	0.4	2	9.35 V	4	Left at marker 03.
(29)	30	0.043	0.43	2	9.35 V	4	Left at marker 03.

Nov 21 2024

	Speed	Kp	Kp	Start Pos	Batt. Voltage	Oscillation Rating	Results:
(30)	30	0.044	0.44	2	9.35 V	5	Left at marker 03
(31)	30	0.0445	0.445	2	9.35 V	5	Left at marker 03.
(32)	30	0.015	0.075	2	9.35 V	5	Left at marker 03
(33)	30	0.02	0.1	2	9.35 V	5	Left at marker 03
(34)	30	0.03	0.15	2	9.35 V	5	Left at marker 03
(35)	30	0.0447	0.447	2	9.35 V	4	Left at marker 03
(36)	30	0.0447	0.2235	2	9.35 V	4	Left at marker 03
(37)	30	0.05	0.25	2	9.35 V	4	Left at marker 03
(38)	30	0.06	0.6	2	9.35 V	4	Left at marker 02
(39)	30	0.06	0.3	2	9.35 V	4	Left at marker 03
(40)	30	0.055	0.55	2	9.35 V	3	Left at marker 03
(41)	20	0.04	0.4	2	9.35 V	3	Left at marker 03
(42)	20	0.01	0.15	2	9.35 V	5	Left at marker 03
(43)	20	0.01	0.05	2	9.35 V	5	Left at marker 03
(44)	20	0.01	0.03	2	9.35 V	4	Left at marker 03
(45)	20	0.01	0.01	2	9.35 V	4	Left at marker 03

Added code for if the ~~adjustment~~ movement command is greater than 0.75 x base speed, one wheel turns off.

(46)	20	0.01	0.05	2	9.35 V		Left at marker 03.

changed 0.75 multipler to 0.6. ~~Added~~ changed the motor that moves to twice its current value.

(47)	20	0.01	0.05	2	9.35 V	5	Left at marker 03.

changed how much the motor changes from 2x → 3x

commented out these multipliers.

(48)	20	0.0005	0.0	2	9.35 V	1	Left at marker 2
(49)	100	0.0025	0	2	9.35 V	1	Left at marker 2
(50)	100	0.05	0	2	9.35 V	2	Left at marker 03
(51)	100	0.025	0	2	9.35V	2	Left at marker 02
(52)	100	0.06	0	2	9.35 V	1	Left at marker 1.
(53)	20	0.01	0.0005	2	9.35 V	2	Left at marker 03
(54)	20	0.02	0.0005	2	9.35 V	3	Left at marker 3
(55)	20	0.025	0	2	9.35 V	3	Left at marker 3
(56)	20	0.03	0	2	9.35 V	3	Left at marker 3
(57)	20	0.05	0	2	9.35 V	2	Left at marker 3
(58)	20	0.04	0	2	9.35V	2	Left at marker 05
(59)	20	0.09	0	2	9.35 V	1	Left at marker 03

Nov 21st 2024

	Speed	K_p	K_D	Start Pos	Battery Voltage	Oscillation Rating	
(60)	20	0.1	0	2	8.98 V	1	Left at marker 02.
(61)	20	0.08	0.008	2	8.98 V	2	Left at marker 03.
(62)	20	0.09	0.009	2	8.98V	3	Left at marker 04
(63)	20	0.1	0.01	2	8.98 V	1	Left at marker 02.
(64)	20	0.13	0	u-turn	9.03 V	2	half of the uturn, no pass
(65)	20	0.12	0	u-turn	9.03 V	1	no pass

Added a section of code where it will reverse one wheel when the |correction value| is greater than $\frac{1}{2}$ of base speeed.

(66)	30	0.05	0	2	8.5 V	4	left at marker 02
(67)	30	0.05	0.005	2	8.5 V	5	left at marker 04
(68)	30	0.05	0.007	2	8.5 V	5	left at marker 03.
(69)	30	0.05	0.001	2	8.5 V	5	left at marker 14
(70)		changed	|correction value| > $\frac{1}{2}$ of base speed to |correction value| > 2 times base speed.		
(70)	30	0.05	0.002	2	8.5 V	5	Left at marker 14.
(71)	30	0.05	0.0015	2	8.5 V	5	Left at marker 14.
(72)	30	0.05	0.0013	2	8.5V	5	Left at marker 24.