Final Report

EE3

William Randall

Talia Wolf-Jacobs

Introduction and Background:

The goals of this project were to implement a car that would follow a line with a gradient path using a closed-loop control system that would keep it on the center of the path as the path curved. We wanted to make the car travel down the path, turn around and come back to the start, and stop again, all in under 45 seconds. We chose a PD control[1] system based on the concepts discussed in lecture, proportional (to respond to the current error term) and derivative (to prevent over correcting and increase stability), which allowed us to read in sensor values and dynamically change the speed of each wheel to keep the car on course. We read in values and weighted each sensor to promote the car to stay in the center of the track. The sensors on the car read in a range from close to zero (complete bright) up to 2500 (complete dark). Colors that reflect light, such as white, produce a low voltage output from the sensors whereas colors that absorb would output high. This is because the NPN phototransistor responds to light coming in the base as described previously by either allowing or preventing current from flowing to ground. The ideal diode equation $i(v) = IS[\exp(v / \eta VT) - 1]$ can help to describe how the current allowed through the phototransistor varies as the base is exposed to varying levels of light. As the car moves off-center, the sensors on the underside of the car would stop reading symmetrically, so we could sum the differences in readings as an error term to tell how off track the car was and in which direction to move, in accordance with the PD weightings. On curves in the track, the sensors read more light and thus lower voltages on the side that the car is turning away from, so the PWM supplied to that wheel is increased to increase the speed and turn the car. The wheels are powered by a Pulse Width Modulation (see figure below) signal, which takes a DC signal and pulsates it at various duty cycles to turn the wheels at various speeds. Additionally, we completed all of the extra credit options, implementing odometry and speed control and computing variance.

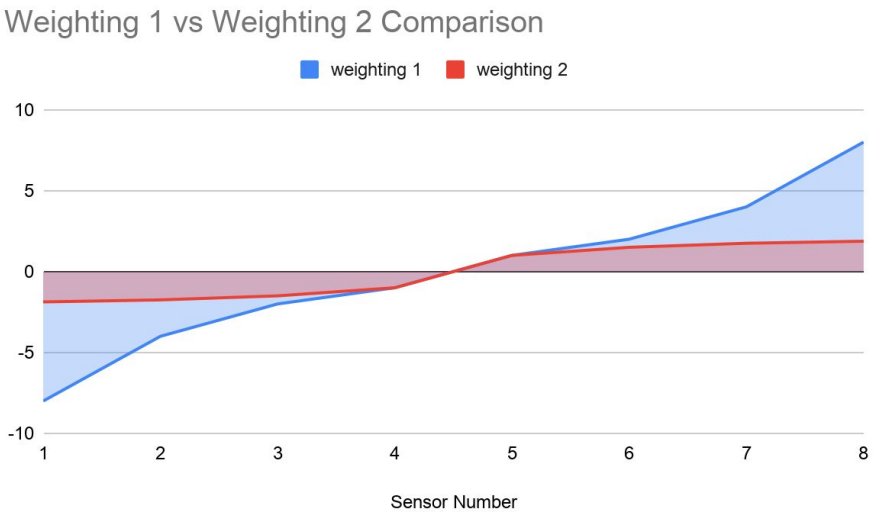


Figure 1.1: The chart above shows the graph of the weightings we used for the sensor values. Weighting 1 is -8, -4, -2, -1, 1, 2, 4, 8 and weighting 2 is -1.875, -1.75, -1.5, -1, 1, 1.5, 1.75, 1.875.

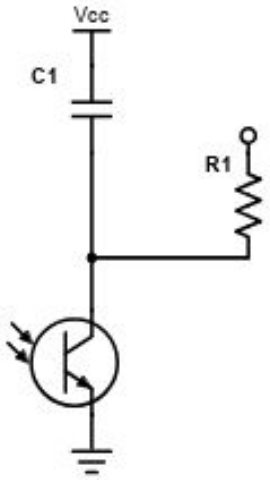


Figure 1.2: The circuit schematic above shows one of the eight sensors on the bottom of the car, with a capacitor of C=10μF, a resistor of R_1=220 kΩ, and phototransistor. Created using Scheme-It [3].

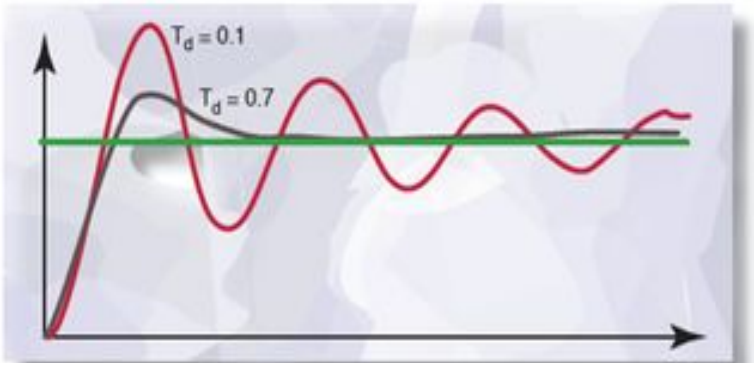


Figure 1.3: An illustration of how the derivative term in PD control helps to reduce oscillations and keep the car on the track without overcorrecting. Avery, Paul. "Introduction to PID Control." StackPath, 1 Mar. 2009, www.machinedesign.com/automation-iiot/sensors/article/21831887/introduction-to-pid-control.

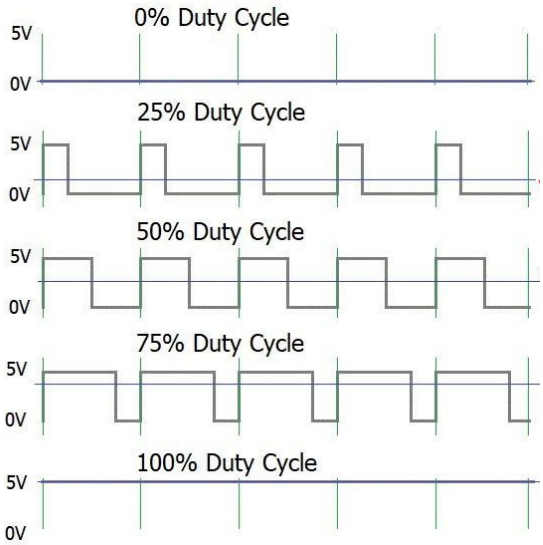


Figure 1.4: Illustration of PWM signal at various Duty Cyle values.
Raj, Aswinth. "What Is PWM: Pulse Width Modulation." *What Is PWM: Pulse Width Modulation*, 19 Sept. 2018, circuitdigest.com/tutorial/what-is-pwm-pulse-width-modulation

Testing Methodology:

First, my team had to verify that the sensors on the bottom of the car were working properly. To do this, we made a program that would print out the raw value of each sensor into the serial monitor. We used this program to watch the values of each sensor as we moved the car back and forth over a part of the track. We observed that when the sensor was over a black part of the track it would measure a value of above 1700 and when it was over white it would read a value of below 500. We made sure that this subsystem was working by making sure that each sensor would read out the correct values in accordance with what we observed from the other sensors.

Test Setup:

For tests of what Kp, proportional constant, and Kd, derivative constant, we tried to separate our test into groupings of base speeds because when we changed the base speeds the Kp and Kd constants would be slightly different. We would use a measured value for our PD controller which was a combination of each sensor multiplied by a weighting. The two versions of the weightings we tested for the 8 sensors was -1.875, -1.75, -1.5, -1, 1, 1.5, 1.75, 1.875 and -8, -4, -2, -1, 1, 2, 4, 8. This would give us a value of the rough integral of the sensor values, which we could use to consolidate the values to use as an input in our PD controller. These values we would get from the sensors are values from phototransistor going into the analog inputs of our car. The values come from the fact that if we put the phototransistor over a white sheet of paper it would give us a low voltage through our analog input and if we put it over a black sheet of paper the reading of our analog input would be high. So after we put our measured value through our PD control it would give a positive output if it needed to turn left and a negative output if it needed to turn the car right.

How the tests were conducted:

To collect data we would start with an equal Kp and Kd value and try it at a certain base speed of our car. Based on the performance of the car at these values we would either increase Kp or Kd. For example, if the car was oscillating too much and dampening of the oscillation seemed like it was not occurring then we would increase the Kd constant, the derivative constant. On the other hand, if the car seemed like it could not stay on the track and it would miss turns we would increase the Kp constant, proportional constant because it would help the car stay on the track. Additionally, if we found that we had a working Kp and Kd but the car was jittering and being too reactive then we would half both the Kp and Kd constant because that would mean that the car was reacting too drastically to changes in the sensor values. We would then qualitatively record the behavior of the car at each of the tested values. The data we collected can be seen in the table below.

Data Analysis:

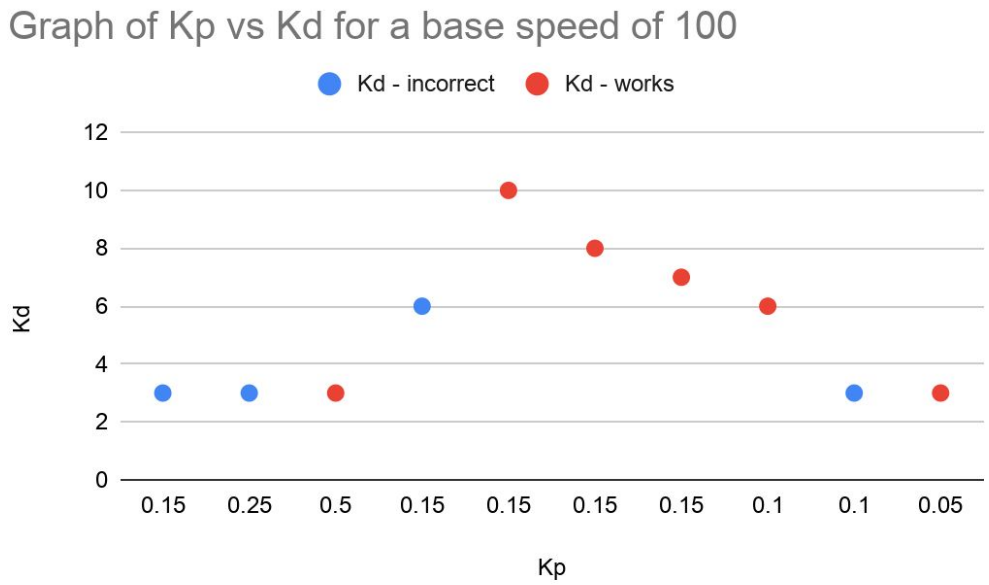
The analysis we did of Kp and Kd was if the car could not follow the track it was due to one of two reasons. One, it would not recognize the track well and would miss turns, this would be fixed if we increased the Kp constant to increase the value we put on the current measured value of our sensors. Two, if the car was oscillating a lot and wasting time moving down the track we would increase the Kd term to dampen the oscillation because the Kd term would compare the current measured value to the previously measured value to effectively straighten out the car as it moved down the track. Also, once we found a working Kp and Kd value at a certain speed we would increase the base speed and use the same Kp and Kd then tune from that point.

Speed	Kp	Kd	Observation
First iteration of code for the car with sensor constants of -1.875, -1.75, -1.5, -1, 1, 1.5, 1.75, 1.875			
90	1.25	1.5	It reacted to turn very slowly
90	1.75	1	It continued to react slowly to the line and miss turns
100	1	1	No reaction to curves in line
100	3	2	It jitters a lot back and forth. It almost worked but it missed the turn.
100	2.25	2	Spiraled and missed the turn
100	2	2.5	Worked for one turn but missed the second turn
100	5	7	Almost finished but missed the last turn
100	7	5	Oscillated and missed the turn
100	5	6	Completely missed the turns and track
100	7	6	It missed the first turn
100	8	5	Got the first turn barely and oscillated a lot
100	7	4	Oscillated a lot then it did not react to the turn
100	6	5	Oscillated fast then missed the first turn
100	4	2.5	Oscillated a lot then missed the turn
100	3	1	Oscillated then missed the first turn
100	1	3	Did not oscillate then missed the turn by veering off the track
100	1	1.5	It says on the track but then veers off

100	1	1.2	It misses the entire first turn
100	1.25	1.5	It misses the entire first turn
70	1	1.5	It makes it almost the entire track
70	1.3	1.5	It made it the entire track but it seems like it might come off the track
70	1.3	1.6	It completed the track but it is very unstable
70	1.4	1.6	It makes it the whole track but is still unstable
70	1.4	1.7	It is better but it still does not dampen the oscillations
70	1.4	1.8	Stays on the Track and finished in 43 seconds but we could improve on the oscillations
Changed the way we had our output was mapped to our analogWrite and we changed our sensor constants to -8, -4, -2, -1, 1, 2, 4, 8			
100	.25	1.5	Oscillates a lot then falls off the track
100	1	1.5	Oscillates a lot then falls off the track
100	1	0.5	Misses the track
100	.25	2	It is better and oscillations decrease
100	.15	2	So fast that it overshoots the final turn so we implemented an acceleration function
100	.15	3	Jittery but it is pretty good at following the track
We changed the constants back to -1.875, -1.75, -1.5, -1, 1, 1.5, 1.75, 1.875			
100	.15	3	Finishes in 22.5 seconds but it oscillates and jitters
100	.25	3	Finishes but still jitters back and forth
100	.5	3	It is a lot worse than before and falls off the track
100	.15	6	Less oscillation and it is faster
100	.15	10	It misses the first turn
100	.15	8	It misses the first turn
100	.15	7	It misses the first turn
100	.1	6	It misses the first turn

100	.1	3	It has fewer oscillations and can finish
100	.05	3	It misses the first turn
110	.1	3	It almost finishes but misses the last turn, which might have been because of the footprints because it is now really sensitive
110	.1	4	It misses the last turn
110	.1	3.5	It finishes in 22 seconds
110	.07	3.5	It finishes
110	.06	3.5	It misses the turn
110	.07	3.75	It misses the last turn
110	.07	3.6	It misses the second turn
110	.07	3.55	It misses the last turn
110	.1	3.4	It does not miss the turn but I did not try the whole track
120	.1	3.5	It does not work very well because of the oscillations
120	.01	.35	It works the best with the least oscillations
120	.001	.035	Car missed the turns
120	.01	.34	It missed the first turn
120	.012	.35	It missed the last turn
120	.014	.35	It missed the last turn
120	.017	.35	It was better but missed the last turn. We think it is because Kp is too low.
120	.02	.35	Works the whole way and finished in 22 seconds
140	.02	.35	It works but we did not have time to see if it finishes the whole track.
160	.02	.35	It misses the last turn
160	.023	.35	It misses the last turn
160	.025	.35	It misses the last turn
160	.03	.35	It misses the last turn
160	.04	.35	It misses the last turn

160	.05	.35	It oscillates a lot and makes the last turn
160	.02	0.175	It oscillates a lot and misses the last turn
160	.01	.175	It does not miss the first turn but then it misses the second turn
160	.01	.15	It does not miss the first turn but then it misses the second turn
160	.01	.1	It completely misses the first turn
160	.015	.175	It completely misses the first turn
160	.02	.175	It oscillates a lot but makes it almost the whole way
160	.02	.18	It oscillates but misses the 3rd turn
160	.02	.15	It seems like it is about to fall off the track but it does not
160	.02	.09	It misses the entire track
160	.025	.15	It oscillates a bunch and misses the track sometimes
160	.03	.15	It oscillates a lot and misses turns
160	.027	.15	It oscillates a lot and misses the track
160	.026	.15	It does not dampen the oscillation
190	.02	.35	It does not work or respond to the track
190	.03	.35	It oscillated then did not follow the track
150	.02	.35	It did not work
150	.03	.35	It did not work
Race Day Track			
140	.02	.35	Makes it the whole way in 18.51 seconds
150	.02	.35	Makes it the whole track in 15.68 seconds. It has a variance of 1.33.
160	.02	.35	Makes it the whole track in 15 seconds.
190	.025	.35	Make it the whole track.
210	.025	.353	Makes it the whole track.
240	.025	.353	Final Speed and Kp Kd. It finishes the whole track in 12.86 seconds.



This graph shows the values of Kp vs Kd and each is colored red if the car completed the track and blue if it did not complete the track.

Test Data Interpretation:

From the data we collected, it seems like because we were testing on the track that was not pristine, we had to have better PD control on our car. This is seen when we were able to increase the base speed on the car on race day when we had the better track, and still did not have to change our Kp or Kd. At higher speeds, we found that the Kd constant must be much higher than our Kp constant and we think this is because the car must look into the future and fix oscillations more quickly because it would lose the track it did not. As we increased to base speeds above 200 our Kp was about 7% of our Kd. We also observed how to tune the PD controller when we saw certain signs such as non-damped oscillation and the complete missing of the track line, increase Kd and increase Kp respectively. For example, when we tried Kp of 0.017 and a Kd of .35 at speed 120 we found that it missed the last turn so we increased the Kp to 0.02 and that fixed the problem because it stayed on the track better. Also when we found that the car was jittering with a Kp of 0.15 and a Kd of 3 at speed 100 we divided both the Kp and Kd by half and the jittering went away. Another factor that we tuned was our turn delay time which made the times slower because if the car did not make a perfect turn of 180 degrees then the car had to catch the track at an angle then dampen the oscillation which took extra time.

Results and Discussion:

Test Discussion:

Our graph and table of data were appropriate to our goal of making a car that used a closed loop control system. It allowed us to record what constants caused what qualitative behavior in our car system. We were able to use a closed loop system to take in sensor data and translate that back into an output for our car's wheels. Our graph helps us visualize the trend of the relationship between Kp and Kd at a certain speed, but if we were to do this experiment again we should take more data points at each speed and plot the relationship between the successful and unsuccessful Kp and Kd combinations. Other things we could have worked on was taking more data using a cleaner track so we could get results that matched the race day results. Overall, our interpretations of the test data guided us to finding the best Kp and Kd combo, which helped us get the best time for our car.

Race Day Discussion:

i) How our car performed on Race Day

hyperlink to fastest time of our car (12.86 seconds)

This video was the video of our fastest time of 12.86 seconds, but if we fixed the turn time we could have decreased the time even more. We could have also increased the base speed to 245 or 250 because our code would have been able to limit each wheel to stay in the range of the analog write, which is 0 to 255.

We did run into one problem which was the turning of our car, and that caused one of our runs to fail. Overall, our car made the fastest time out of the entire class, and we had a final variance of 1.33. We also were able to do the odometry bonus and we were able to turn on the onboard led at the correct point on the track.

We were able to do all three bonuses on race day. The closed loop speed control bonus we had used a PID controller where we introduced an integral term to compensate for the friction that would be applied to test this bonus. It was able to maintain a certain speed even when friction was applied. We were also able to complete the variance and the odometry bonus on race day. The odometry bonus worked on our first try and the led illuminated about ¾ of the way down the track after it performed its turnaround, but we turned that function off when we tried to get the lowest time. Finally, the variance bonus gave us a low variance of 1.33 the first run we did it on, and we implemented the variance function by making an iterative variance calculator that ran the entire time the car was running then only displayed after the car stopped when it made a complete run.

ii) Limitations of our Code and of the Car and Tracks

There were a number of limitations presented by our code and the tracks. Even at our fastest time, the car still had turning time that could have been reduced if we used a different turning method. Also, our code accelerated at the start, so we could potentially make the ramp up time shorter so that us us just slow enough that the car does not skid, that would make the car faster. Mechanically, the wheels can only spin so fast and as the speed increased, they were often not heavy enough to avoid skidding. The track itself provided limits as well because the turning radius on the turns was very small, so at a certain point the car has to overcome its own inertia and that strains the motors.

iii) Changes if We Redo This Project

Given what we learned while doing this project, there are several changes we would make. We would implement a different, more efficient braking system that stops the car without any turning of the car with another function that would read in the encoder values and make sure that it stops without turning. Additionally, we would write another PD controller that looks at the encoder count of each wheel and precisely makes a 180 degree turn, to eliminate the guesswork of the turn timing and prevent overshoots. These changes would speed up our car by making our turn more efficient and produce a more elegant design.

Conclusions and Future Work:

Our project successfully completed the track in well under 45 seconds, and the design improved drastically from the initial version to the final race day iteration. We learned the basics of PID control implementation and gained experience implementing the design principles we learn in our classes. Additionally, the extra credit projects allowed us to explore using displays and computing useful metrics (like variance). If we had more time, it would be interesting to try different form factors and weights for the car to see how the traction changes the timing and algorithms. We could test this by using different proportional and derivative constants with the heavier car and testing how the timing and control systems respond. Another extension we would like to explore would be to use a more complicated track with physical obstacles and to use proximity sensors on the car to avoid those. To test this, we would have to combine our code with an obstacle sensing algorithm and gauge the sensitivity and outputs of the obstacle sensors with a similar methodology to how we did this project with the light sensors.

References:

1. Avery, Paul. “Introduction to PID Control.” *StackPath*, 1 Mar. 2009, www.machinedesign.com/automation-iiot/sensors/article/21831887/introduction-to-pid-control.
2. Notes, Electronics. “What Is a Phototransistor: Tutorial & Primer.” *Electronics Notes*, www.electronics-notes.com/articles/electronic_components/transistor/what-is-a-phototransistor-tutorial.php.
3. “Scheme-It.” *Scheme-It | Free Online Schematic and Diagramming Tool | DigiKey Electronics*, www.digikey.com/schemeit/project/.
4. Raj, Aswinth. “What Is PWM: Pulse Width Modulation.” *What Is PWM: Pulse Width Modulation*, 19 Sept. 2018, circuitdigest.com/tutorial/what-is-pwm-pulse-width-modulation.