# EE3 Final Report: PD Controlled Path Following Car

ECE 3, Spring 2022, June 4 2022

**Eugene Min, Utkarsh Lal**

## Introduction

The goals of this project were to implement a car that would follow a line with turns and curves using a  loop control system that would keep it on the path as the path curved. The car goes through a curved line path and makes a donut turn at the end to retrace its path and finally stops after reaching the starting point. We used a PD control system based on the concepts discussed in the lectures:  proportional and derivative, which allowed us to read in sensor values and change the speed of each wheel to keep the car on course.

Colors that reflect light, such as white, produce a low voltage output from the sensors whereas colors that absorb, here black track, would output high and this concept is illustrated in Figure 1. As the car moves off the black track, the sensors on the underside of the car would read in the voltage outputs, sum the differences in readings as an error term to tell how off track the car was and in which direction to move.

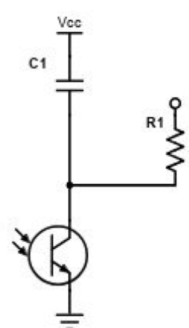


Figure 1 : Circuit of one of the eight sensors. It functions using a capacitor, resistor, and phototransistor.

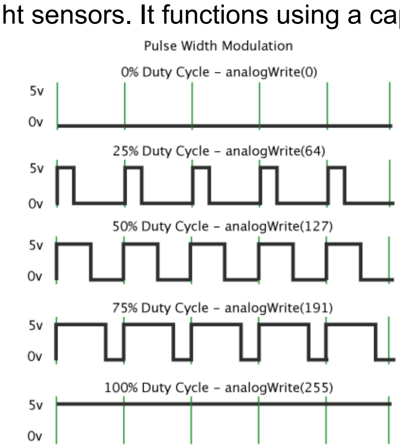


Figure 2: PWM Signals: How speeds correspond to a signal

The motors on the car were controlled by a PWM (Pulse Width Modulation) signal that can range from 0-255. This will be the range of the speeds that we discuss in this report. A PWM signal is a square wave that helps the car determine its value based on the signal's length and ultimately duty cycle. This can be seen in Figure 2.

## Conducting Tests

We controlled the Kp - proportional constant, and Kd - derivative constant by testing its different values in the car. We also controlled the base speed of the left and right wheels.

We measured the encoder count on specific portions of the track. Using that data we controlled where we needed to change the speed of the car on the track. For example, the speed on the straight path can be much larger than the one for sharp turns so we measured the encoder count when the car reaches the straight path of the track and coded it in Arduino to change the speed as soon as it hits that count. We also measured the calibration of all 8 sensors of the car.

## Analysis

Handwritten data observations can be found on the link given below.
[https://drive.google.com/drive/folders/10jQ5AXTRwpxd4Ezw66sXXKoGqlf-RB26?usp=sharing](https://drive.google.com/drive/folders/10jQ5AXTRwpxd4Ezw66sXXKoGqlf-RB26?usp=sharing)

| Speed | Kp | Kd | Observation |
|---|---|---|---|
| 70 | 0.04 | 0.2 | Almost completed the second turn |
| 70 | 0.04 | 0.25 | Completed one full track fell off after doughnut |
| 70 | 0.01 | 0.05 | Track completed in more than 30 seconds |
| 100 | 0.01 | 0.15 | Worked on the turns, failed on straight |
| 100 | 0.019 | 0.17 | Completed the tracks in ~25 seconds |
| 100 | 0.025 | 0.075 | Oscillating too much on the curve and losing the track |
| 100 | 0.025 | 0.08 | Unstable with new batteries in the car |
| 100 | 0.025 | 0.1 | Takes a diagonal path across two quick turns and unstable in straight line |
| 170 | 0.009 | 0.02 | Unstable on straight track |
| 170 | 0.007 | 0.03 | Almost works on straight path after several tries |
| 170 | 0.0073 | 0.0029 | Works on a straight path inconsistently. Fails on curve. |

Table 1: Sampling of Raw Data

The sampling in Table 1 demonstrates the variability in what we could change. We can also view interesting observations that we saw. For example, changing the batteries would power up the motors causing instability with the values we had because the car would need lower KP to offset the increase in speed. Another interesting one comes in examining which sections the car could and could not complete.

For the chicane,

| Speed | Kp | Kd |
|---|---|---|
| 70 | 0.01 | 0.05 |

For the straight line,

| | | |
|---|---|---|
| 170 | 0.0073 | 0.029 |

For the curve,

| | | |
|---|---|---|
| 100 | 0.019 | 0.17 |

Table 2: Final Values used on Race Day

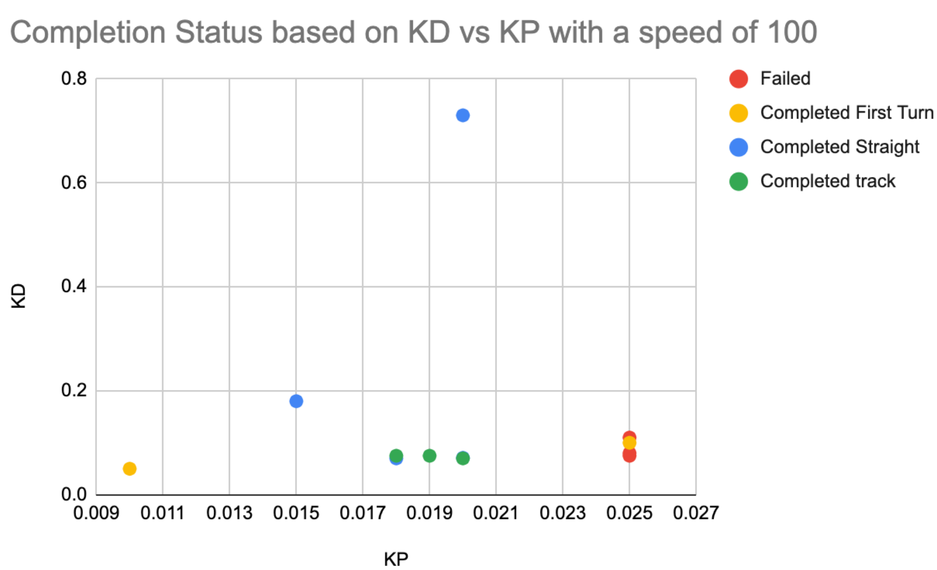


Figure 3: Graph of Kd vs Kp for a base speed of 100 and their observations

We decided to make a graph for speed 100 because before implementing speed changes, we planned on using 100 as a base speed for completing the whole track without speed changes. We tested these KP and KD values to determine the most optimal numbers. As you can see by the legend in Figure 3, each value is denoted with how successful it was.

When the KP was really big relative to the KD the KD could not correct the KP changes fast enough, so the car could not go straight meaning it failed the track completely, denoted by red.

Observing the yellow values, which indicate completion up to the first turn, they only appear at two extremes, when the KP was low or high and KD was relatively low, which makes sense because the car would not have enough KD correction on the straight meaning the car would oscillate off the track.

Looking at the blue values that indicate completion up to the straight, we see that when the KD is too high at a certain point, the car wants to stay on the track too much, meaning that it had trouble on the large curve because it kept wanting to maintain its path.

The cluster of green is indicative of how we were able to narrow down our KP and KD to a working range of KP and KD values, which is around .017 to .021 for KP and .1 for KD.

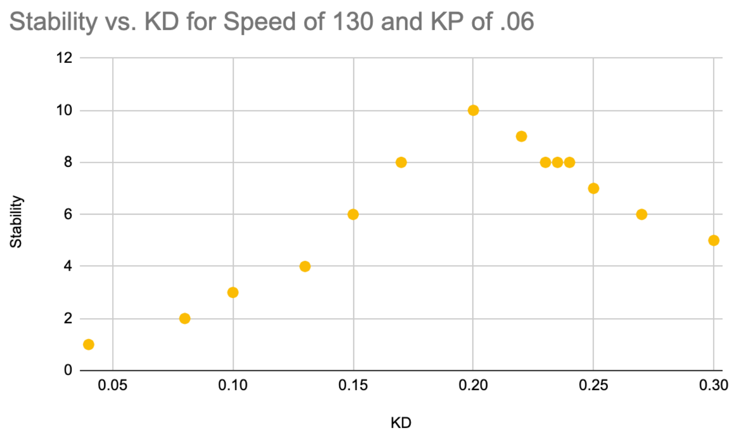


Figure 4: Stability vs Kd plotted for a speed of 130 and KP of .06

In Figure 4, we define stability as how much the car oscillated on the track with a constant KP of .06, where 10 is the least amount of oscillations and 1 as the most oscillations. Looking at the general shape tells us that when the KD is too low, there is not enough correction leading to high oscillations. When the KD is too high there is overcorrection, which also causes further instability. The high KD values are related to the behavior we saw with the blue data points in Figure 2, where the car wants to stay on the track so much that the car will not be able to follow a curve and oscillate a lot believing it is correcting itself sufficiently.

Conclusively, Figure 4 defines the most optimal KD to be .2 for a KP of .06 and a speed of 130.

## Interpreting Tests

We will be interpreting a test run where we ran the car with a *speed of 100, a KP of 0.01 and a KD of 0.15*. We observed that it would work on the turns however it failed on the straights. We characterized this run with a stability score of 6 because it oscillated too much and came off the track, but could still complete the track.

One implication of this run is that turns and straights are two different things for the car and that methodically, when the car is faced with specific turns or straights, we want to use varying KPs and KDs. However, it is maintained that turns are more similar to straights no matter the turn's angles from the car's PID control perspective.

The inability of the car to follow the straight can have two implications. One implication is that the KP is so high that it causes too many oscillations in the straight. The other implication could be that the KD is too low so it will not properly offset the car on the straights.

Lastly, another implication of this test is that it is confirming that our KD to KP ratio should be quite large. In this case it was a 15:1 ratio, and this informs us of how we should consider further tests, which will be further elaborated in the conclusion.

## Conclusion

We believe that the in the case of this test run given in the Interpret test section, it is a KD issue because the KP change is relatively low considering that we would only be changing the speed by .01*800 = 8. In addition, past tests without KD and with KD demonstrated significant decreases in oscillations in tests that included KD indicating that KD is the solution to fixing oscillations on the straight. Moreover, past tests were demonstrative that the KP of .01 was sufficient for the turns.

Thus, we decided to change the KD. We increased the KD to 0.17 keeping in mind that we still want to keep the KD:KP ratio high. A KD of .17 improved the performance on the straight track but was unsuccessful on the turn indicating the car's unwillingness to deviate its path. Consequently, we decided to increase the KP to 0.014 to make the car responsive to the turns, which resulted in the turns not consistently succeeding still. Through this method of adjusting the KD and KP one after the other, we reached a conclusive KP and KD of .019 and 0.17 illustrated in Table 2 for the speed of 100.

Our car completed the track in well under 18.5 seconds which earned us extra credit for completing it under the 20 second threshold. Our performance improved significantly from the initial version to the race day after implementation of speed changes for different portions of the track and a proportional (KP) and derivative constant (KD). We learned the PID control implementation and gained experience implementing the design principles we learn in our classes.

If we had more time, we would have tried more constant value pairings to try increasing the speed along the straight path and curve. We could've also created a system to detect and safely avoid phantom crosspiece. Another challenging extension to the project could have been to find a diagonal path across two sharp turns on the path.

Race day car video-
https://drive.google.com/drive/folders/1--yjJ3xOyAUz6OgCm5c_NqxpclfR1MyG?usp=sharing

## Acknowledgments

We would like to extend our sincere thanks to Dr. Dennis Briggs who made this incredible project possible. His guidance, resources and knowledge carried us through the fundamental concepts of circuits theory, PID control development and a history of electrical engineering. Teaching assistant Dhruv Srinivas alongwith Dr. Briggs, helped us learn foundational laboratory skills along with experience with laboratory equipment. This project is a strong implementation of real-time embedded systems programming.

## References

1. O. M. Stafsudd, Rev. Dr. Dennis M. Briggs (September 2020). “EE3 Introduction To Electrical Engineering Laboratory Manual.” 2013.
2. Scott Zhuge. “PID Control Theory.” *Crystal Instruments*, August 24, 2020, https://www.crystalinstruments.com/blog/2020/8/23/pid-control-theory
3. Daniel de Waard. “Arduino Pulse Width Modulation.” *SuperHouse,* February 16, 2017 https://www.superhouse.tv/arduino-pulse-width-modulation/

## Appendix

| | Speed | $K_p$ | $K_d$ | Comments |
|---|---|---|---|---|
| 1 | 75 | 0.01 | 0 | Small oscillations but turns. |
| 2 | " | " | 0.01 | jitters on straight path |
| 3 | " | 0.009 | 0.05 | Falls off the track immediately |
| 4 | " | " | 0.03 | A little stable than previous but still falls off. |
| 5 | " | 0.01 | 0.02 | " " |
| 6 | " | 0.008 | 0.02 | Very unstable |
| 7 | 100 | 0.008 | 0.01 | " |
| 8 | 75 | 0.008 | 0.01 | Smooth on curves. Missed the 180 spin. |
| 9 | 85 | 0.008 | 0.01 | Unstable on track; falls. |
| 10 | 80 | 0.008 | 0.25 | Misses 2nd turn |
| 11 | " | " | 0.2 | Doesn't turn at all. |
| 12 | " | 0.01 | 0.24 | " " |
| 13 | " | 0.011 | 0.23 | Almost completes 2nd turn. |
| 14 | " | 0.014 | 0.22 | " " |
| 15 | " | 0.018 | 0.2 | " |
| 16 | " | 0.025 | 0.2 | " |
| 17 | " | 0.04 | 0.2 | " |
| 18 | 70 | 0.04 | 0.2 | " |
| 19 | 60 | 0.04 | 0.2 | " |
| 20 | 60 | 0.016 | 0.2 | |
| 21 | 70 | 0.01 | 0.05 | Fell off first curve. |
| 22 | 170 | 0.009 | 0.02 | Track completed in 28sec. |
| 23 | " | 0.009 | 0.022 | Unstable on straight path |
| 24 | " | 0.008 | 0.022 | Misses the second turn |
| | | | | Worked on turns, failed straight path |

| | Speed | $K_p$ | $K_d$ | Comments |
|---|---|---|---|---|
| 25 | 170 | 0.0078 | 0.03 | Almost works straight |
| 26 | " | 0.0073 | 0.029 | Works on the straight path; fails on curve |
| 27 | 100 | 0.02 | 0.07 | Failed at 180 spin because unstable |
| 28 | " | 0.025 | 0.075 | Oscillating too much |
| 29 | " | 0.025 | 0.08 | Unstable with new batteries |
| 30 | " | 0.025 | 0.11 | Unresponsive to curve |
| 31 | " | 0.015 | 0.18 | Misses last turn |
| 32 | " | 0.019 | 0.17 | Completes track in ~26 sec |

For race day, we decided to use three different PID values.

1. For sharp turns in beginning,
   (21) Speed = 70 ; $K_p$ = 0.01 ; $K_d$ = 0.05
2. For straight path,
   (26) Speed = 170 ; $K_p$ = 0.0073 ; $K_d$ = 0.029
3. For curves,
   (32) Speed = 100 ; $K_p$ = 0.019 ; $K_d$ = 0.17

Completed in 18.5 seconds.

| | Speed | Kp | Kd | Comments |
|---|---|---|---|---|
| 33 | 130 | 0.075 | 0.27 | Unstable on straight |
| 34 | " | 0.06 | 0.27 | " |
| 35 | " | " | 0.3 | " |
| 36 | " | " | 0.24 | " |
| 37 | " | " | 0.22 | Missed the second turn |
| 38 | " | 0.055 | 0.22 | Fails at straight path |
| 39 | " | 0.05 | 0.22 | Turns left falls off afterwards |
| 40 | " | 0.045 | " | Fails second turn |
| 41 | " | 0.048 | 0.23 | Unstable on straight |
| 42 | " | " | 0.24 | " |
| 43 | " | " | 0.235 | " |
| 44 | " | 0.049 | 0.229 | Works on curve only |
| 45 | " | 0.049 | 0.226 | Falls off immediately |