PID Controller

Patrick Cleeve

Objective

The objective of this project is to code and tune a PID controller to be drive the vehicle around the simulator track.

The project code is available here:
https://github.com/patrickcleeve/CarND-Term2-PID-Control-Project-P4

A video (split due to size) showing successful completion is available in the project repository.

PID Control

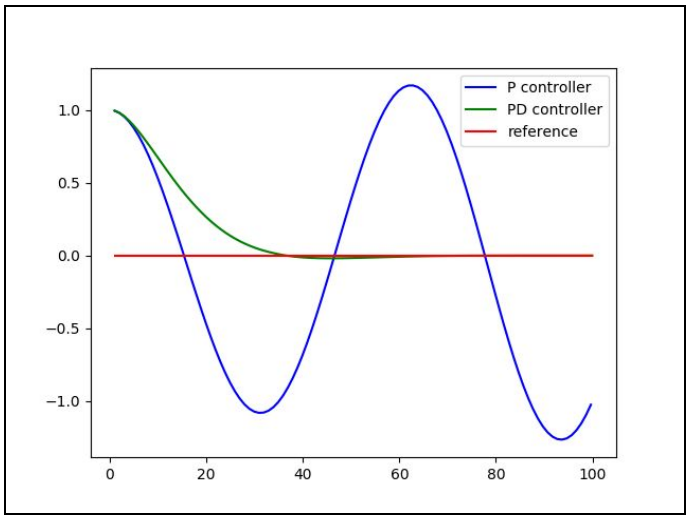
In this project we tune a PID controller based on the Cross Track Error (CTE) for the vehicle driving around the simulator track.

Proportional (P) Component

Proportional control is based on cross track error (CTE) for the current timestep. We can tune this controller using the constant Kp, to determine how much we want to turn in response to the current error. However, sole proportional control will also cause the controller to overshoot and oscillate around the reference. This is because even when the CTE becomes small the controller directs the vehicle across the reference line, causing it to overshoot and continue to oscillate (seen in the image below).

Derivative (D) Component

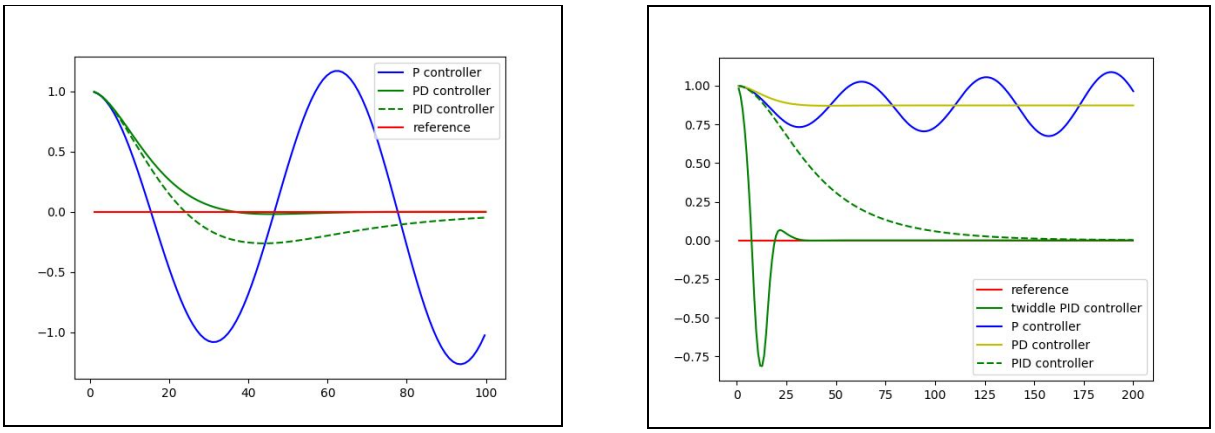
Derivative control is based on the rate of the change of the cross track error (i.e. the difference between CTE at subsequent timesteps). We can alter the constant Kd to tune the controller to damp the oscillations caused by proportional control, and prevent overshooting. In the ideal case, we wish to critically damp the system, so the controller returns to equilibrium in the minimum amount of time (see image). If the damping is too low, the controller will continue to oscillate, whilst if too high it will take too long to reduce the CTE (i.e. respond to changes).



PD Controller (Udacity Lecture)

Integral (I) Component

Integral control is based on the build up of cross track error (CTE) over time (i.e. the integral of CTE). This build up of error occurs because of bias into the controller or system, such as a misaligned steering axle. Without controlling for this build up of error over time, the system cannot reduce this bias and will never reach the reference value. The constant Ki can be used to tune the controller to respond to the build up of this error. The image below shows the effect of bias on a controller, and the impact of integral control.



PID Control without bias and with bias (Udacity Lecture)

Parameter Tuning

The controller parameters were tuned manually, by observing the impact of changing the components on the vehicle performance in the simulator. The tuning process is documented below, including the outcomes of each test. (NB: actually controller values are negative in code).

Kp	Ki	Kd	Throttle	Result
0.5	0.0	0.0	0.3	Initial controller, code is working. Oscillates rapidly and runs off the road. Need to increase D and reduce P.
0.5	0.0	0.5	0.3	Improved distance, still turns too quickly
0.25	0.0	0.75	0.3	Much better, goes off near the bridge.
0.25	0.0	0.75	0.3	Makes it around the track, but drives off road and oscillates
0.2	0.0	0.8	0.3	A bit smoother, having trouble on hard turns. crashes near bridge.
0.2	0.0	0.85	0.3	Similar as above
0.2	0.0	0.9	0.3	Better, still very erratic.Makes it around the track but goes off road. Going to add small integral term.
0.2	0.01	0.9	0.3	Terrible, immediately drives straight off road.
0.2	0.5	0.9	0.3	Same, turns too hard. Going to revisit integral later.
0.2	0.0	0.9	0.2	Success! Still bit wobbly but completes a lap of the track. Stable controller with only PD at lower speed.

The final controller was able to successfully complete a lap of the track using only PD components, and reduced speed (throttle). As the simulator did not have a large bias build up, the error was able to be controlled without integral components. However, this controller would not likely generalise very well to the real world due to these assumptions. Therefore, it will need significant improvements in the future.

Reference

Robotics (Stack Exchange), What arer good strategies for tuning PID loops?
https://robotics.stackexchange.com/questions/167/what-are-good-strategies-for-tuning-pid-loops

Udacity, PID Control
https://classroom.udacity.com/nanodegrees/nd013/parts/40f38239-66b6-46ec-ae68-03afd8a601c8/modules/f1820894-8322-4bb3-81aa-b26b3c6dcbaf/lessons/1397890f-71c5-4d83-aae7-0a031eedd1f2/concepts/c369a13e-1dfa-4a14-a49e-3d04c499a921

Udacity, Self-Driving Car Project Q&A | PID Controller
https://www.youtube.com/watch?v=YamBuzDjrs8

Udacity, Project Slack (#s-t2-p-pid)
No link available

Wikipedia, Damping Ratio
https://en.wikipedia.org/wiki/Damping_ratio