PID Controller Project

Udacity Self Driving Car Nano Degree, Term 2, Project 4

Project

The goal of this project is to build a PID (Proportional, Integral, and Derivative) Controller in C++ and drive a vehicle in a simulator environment. The PID controller determines the steering angle of the car and passes it on to the simulator which applies the steering angle and drives the car around a track. In addition, the PID Controller also controls the throttle of the car.

Project Steps

- Build the PID controller.
- Tune the hyper parameters of the PI controller.
- Test the solution on the simulator.

Reflection

The PID Control lessons explained the individual aspects of the PID controller. These lessons were referred to often while implementing the PID Controller.

- Proportional (P)
  The Cross Track Error (CTE) is defined as how far the car is, from the middle of the track. CTE is a measure of the error and is an input to the Proportional part of the controller. The output of the controller is proportional to the value of CTE. Please note that only the current value of CTE is considered for determining the correct steering angle. If the coefficient for P, Kp is set too high, the car will oscillate. If Kp is set too low, the car will react slowly on sharp curves and will get off the road.

- Integral (I)
  The Integral part of the PID controller accounts for the sum of all the CTE values up to that point in time. The motivation for this is to reduce the bias in the system. If the car is continuously driving on either the left or the right side of the road, the Integral CTE value goes up and makes the car to move towards the center. If the coefficient for I, Ki is set too high, the car will have quicker oscillations. If Ki is set too low, will cause the car to drift to one side of the road.

- Derivative (D)
  The Derivative part of the controller, consider the rate of change of CTE (derivative). If the derivate is changing fast (like on a sharp curve), the car will correct itself quickly. If the derivative is changing slowly (straight road, car is at the center), the steering is smoothened out. If the coefficient for D, Kd is set too high, it leads to a constant steering angle. A low value of for Kd will make the car to oscillate.

Tuning the Hyper parameters

Multiple strategies are available to tune the hyper parameters of the PID Controller. These include using the Twiddle algorithm described in the PID Control, SGD (Stochastic Gradient Descent), manual tuning, or something else, or a combination. I decided to uses the manual tuning, so that I can firsthand experience and understand the effect of the hyper parameters on the performance of the car around the track. After experimenting with multiple values for Kp, Ki and Kd I decided to use Kp = 0.15, Ki = 0.01 and Kd = 3.5 for the final submission. More details about the tuning are provided below.

Manual Tuning

- Proportional coefficient, Kp
  I started with a value of 0.35 but the car experienced very wide swings on the road and never stayed at the center of the track. Reducing the value to 0.05 didn't help either since the car went off the track on a turn. Finally, a value of 0.15 provided consistent results even after multiple turns around the track.

- Integral coefficient, Ki
  I started with a value of 0.11 but that caused very quick oscillations. Values closer to 0 in combination with values of Kp and Kd provided much smoother performance. The final value selected is 0.01

- Derivative coefficient Kd
  I started with a value of 0.004. This value provided wide swings by the car on the track. Increasing the value to 5 made things better and the car stayed on the track. After few more attempts to find the lowest good vale, I finally I settled on a value of 3.5

This behavior of the car on the track for the various values of Kp, Ki and Kd was as expected.

Project Video

PID-Car-video.mp4 has been uploaded to the Github repo. This video is a brief capture of the performance of the car on the simulator track.