PID Controller for Steering

The goals/steps of this project are the following:

- Implement a PID controller in C++ to maneuver the vehicle around the track!
- The simulator will provide you the cross-track error (CTE) and the velocity (mph) in order to compute the appropriate steering angle.
- The project should satisfy the project rubric.

Introduction to PID Controller

A PID controller stands for Proportional Integral Derivative. It is among the most widely used controller due to its ease of implementation and effectiveness. The Proportional term, as the name suggests is used to apply a control action that is directly proportional to the error. But the proportional controller alone can result in overshooting. The differential term in the controller helps in reducing the overshoot by predicting the future behavior of the system. The Integral component of the controller captures the history of the system by integrating all the errors up the point. In case of persistent steady-state error, the integral error keeps on increasing, thereby increasing the control signal and driving the error down.

PID Tuning Approach

- For the project, the PID parameters were manually tuned.
- First, only proportional control was applied, and the Kp gain was set to 1 to see the effect of proportional control. It was seen that the controller was sharply turning the vehicle to the left and right, and the error kept on increasing resulting in the vehicle swerving off the road. (Video: CarND-PID-Control-Project/Video/ P Controller.mp4)
- The Kp gain was reduced to counter the aggressive turning behavior and I finally chose a value of 0.2 for the Kp gain.
- Next, I added the Kd gain term to implement a PD control. The Kd was also set to 1 initially. The PD controller was able to reduce the overshoot caused by the proportional term, and the vehicle was able to stay within the road boundaries. To further reduce the overshoot, I finally chose a value of 2.0 for the Kd gain.

  (Video: CarND-PID-Control-Project/Video/ PD Controller.mp4)

- Then, I added the Ki gain term to implement a PID controller. The Ki gain was initially set to 1. This caused the vehicle to apply a continuous steering input, causing the vehicle to turn in circles. As the vehicle didn’t have any systematic bias, the Integral term wasn’t didn’t help in improving the control behavior. The Ki gain was finally set to 0.001.

  (Video: CarND-PID-Control-Project/Video/ PID Controller.mp4)

Output

The final parameter values for the PID controller were as follows:

- Kp : 0.2
- Ki: 0.001
- Kd: 2.0

(Output video: CarND-PID-Control-Project/Video/ PID Controller Tuned.mp4)