Reflection on PID Controller Project

Outline

In this short report, I summarize the following two points related to my implementation for the PID controller project: (1) the roles of the P, I and D components of the PID controller, (2) hyper-parameter tuning for the PID controller. Note that all the movies refereed here are stored in the accompanied folder named `video`.

(1) Roles of P, I, D Components

Here I explain individually how the P, I, D components of the PID controller affect the control of the steering angle.

- **P Controller**
  The P component controls the steering angle such that the angle is proportional to the cross-track error (CTE) and the car can move toward the center line. The farther the distance between the car and the center line is, the larger the steering angle becomes so that the car can recover to the center line quickly. The downside of this component is that it can cause the overshoot (i.e. go, for example, from left side of the center line to the right side) and lead to a fast oscillation when the coefficient is large. An example of the oscillation due to this component can be seen in the video `https://youtu.be/EQ3GwYZ_sBE` (in this video only the P component is turned on).

- **D Controller**
  The D component controls the steering angle such that the angle is proportional to the derivative of CTE. This component helps to decrease the overshoot and stabilizes the trajectory. An example of the stabilization due to this component can be seen in the video `https://youtu.be/4fS6VkwYQWE` (in this video the P component is the same as the above and, on top of this, the D component is turned on).

- **I Controller**
  The I component controls the steering angle such that the angle is proportional to the integral of CTE over time. In case there is a systematic bias (for example, the tires of the cars are equipped with a tiny but nonzero angle), with the controller with the P and D components only, the trajectory of the car can converge to a line different from the center line. This I component helps to decrease the effect of this systematic bias and let the trajectory converges to the center line. In the video `https://youtu.be/SGb8fDeFqDs`, only the P and D components are turned on and a bias is added for the steering angle. The trajectory of the car is stabilized but not at the center line because of the bias. In the video `https://youtu.be/f8KuIXLPmng`, the I component is added. Thanks to this extra component, the trajectory is stabilized to the center line of the road.

(2) Hyper-Parameter Tuning

In this part, I explain how I fixed the hyper-parameters for the PID controller in my implementation. Note that the throttle value is fixed (=0.3) in Step 1 and 2 below.

- **Step 1: By Hand for Coarse Tuning**
  I first tuned the hyper-parameters by hand such that the car can drive somehow stably around the middle of the road. The coefficients are tuned by taking into account the role of each component explained above.

- **Step 2: Twiddle for Fine Tuning**
  Once I found a set of the hyper-parameters realizing a somehow stable trajectory, next I used the twiddle to tune the hyper-parameters more in detail. In the end, I set the coefficients of the P, I, D components (denoted as `Kp`, `Ki`, `Kd`, respectively) to be the following fixed values: (`Kp`, `Ki`, `Kd`) = (0.13, 0.001, 3.1). The result with these hyper-parameters is recorded in the video `https://youtu.be/yoOLjNrRskI`.

- **Step 3: Throttle**
  To speed up the car, as a final step, I also controlled the throttle value. I set the default throttle value to 0.5 (thus the car can go faster than the case of Step 2) and tuned such that the throttle value decreases as the steering angle goes away from zero. The result is recorded in the video `https://youtu.be/5UQZdofBi6Y` (note that the hyper-parameters (`Kp`, `Ki`, `Kd`) are the same as Step 2). I also considered to build another PID controller for the throttle (on top of the one for the steering), but the performance was not improved very much.