PID Controller
Project 3

Sarbjeet Singh
May 19, 2019

What The PID Controller Does

To achieve the car reaching its target trajectory, the three components of the PID controller play the following role.

The proportional controller takes the cross track error (distance the car center is from the target trajectory) and multiplies it by some constant coefficient to increase/decrease its value. This value is equated to the amount the steering needs to move from zero (car going straight) with a positive or negative value representing moving right or left. The role of the proportionality constant is to determine by what magnitude the steering will move given an error. A larger value enables the steering to adjust faster, but it can also cause large oscillations if the car is constantly over correcting. Small values will cause the car to take longer to adjust and thus not react fast enough. Even a well tuned proportional controller will always have oscillations. An example of a small constant, a good constant and large constant are shown below.

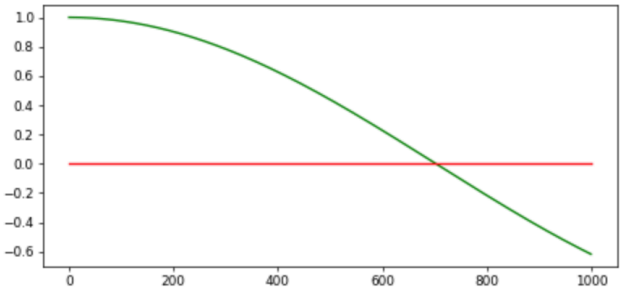

Figure 1: Proportional coefficient is small

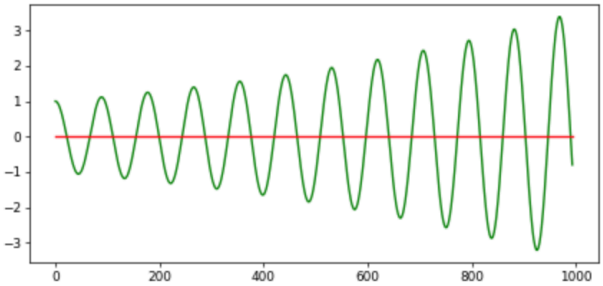

Figure 2: Coefficient is not to large or to small

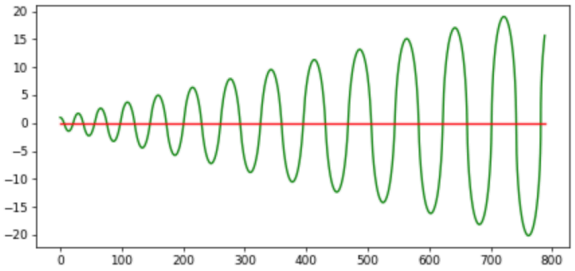

Figure 3: Coefficient is large and oscillation grows quickly

To reduce the oscillations, a derivative term is needed. When the steering direction is such that the car getting closer to the target trajectory, the rate of change is negative (-derror/dt). Thus adding this to the error value will reduce it and the opposite will happen if the rate of change is positive. Multiplying the derivative term by a constant helps to increase or decrease this dampening effect. However a large enough constant can cause the rate of change to over-correct and lead the vehicle away from the target trajectory. For example, if the error is 4 and the rate of change in error is -1, the steering value would than be 4+(-1) = 3 but if the derivative constant is 5 you have 4 +5*(-1) = - 1, which would lead the car away from the target. Examples of a well-tuned derivative function and over-tuned one are shown below.

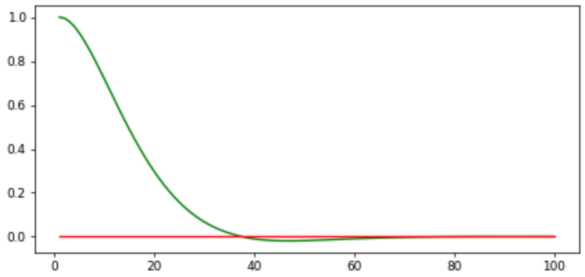

Figure 4: Well-tuned PD Coefficient

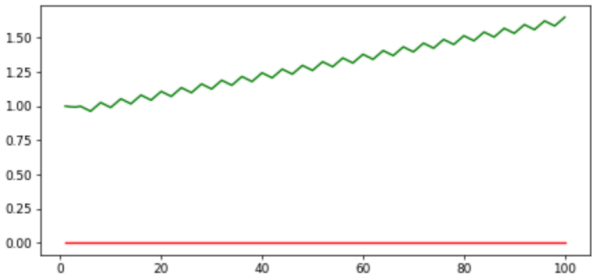

Figure 5: Coefficient is too large

The integrative term is important when forces that can shift the vehicles away from the target trajectory affect an error rate. For examples a steering drift will always cause the car to move away, and the P or D terms will not be able to adjust for this, causing an offset. An example of this is shown below.

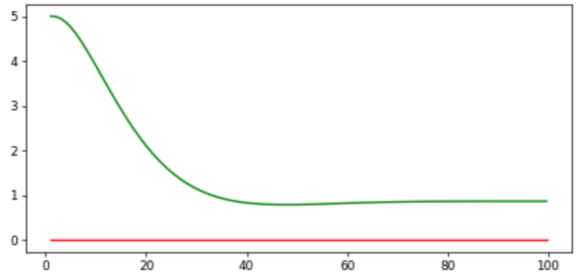

Figure 6: PD controller effected by steering drif

The integral term always adds to the total error and is the summation of all error terms. As the car gets closer, the growth of this value will decrease but essentially it helps remove the offset caused by irregularities in the cars steering. It can be multiplied by a constant term to control the effect of this term. A large integral term will cause the steering to be pushed to its limits in either direction (resulting in a circle as shown below), while too small a value will increase the time to reach the target. A well-tuned coefficient will negate the effect of the error causing parameters.

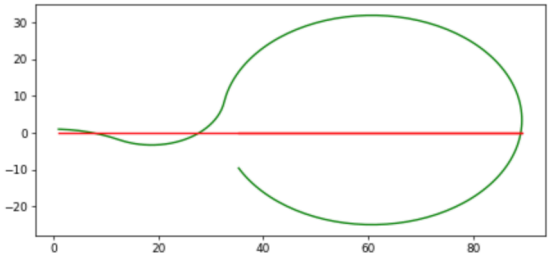

Figure 7: Integral coefficient is too large

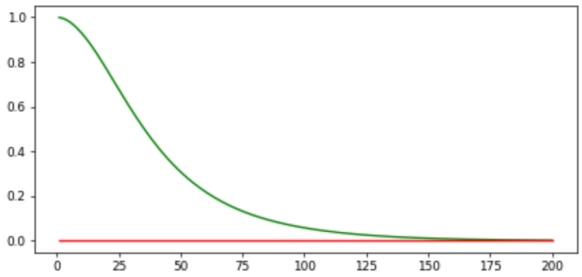

Figure 8: Well-tuned integral coefficient

Selecting Tuning Parameters

I chose to manually tune the PID coefficients for this project. I started with the coefficient values from the lecture (P = 0.2, D = 3.0, I = 0.004) which enabled the car to complete the track successfully when the throttle was set to 0.3. I decided not to utilize twiddle for two reasons. First, the twiddle values that were generated by the algorithm implemented in the lesson were not effective at all in the simulator. Secondly, the twiddle function from the lecture showed me that a car operating with twiddle determined coefficients would overshoot the trajectory at least once, and the values given by Sebastian already performed better than this.
To increase the speed of the car, I also implemented a PID controller for the throttle. I set the coefficients for this to be initially to zero. To tune I followed the following steps while testing the car in the simulator:

1. Start with the proportional value and increase to the point where the oscillations were steady.
2. Start increasing the derivative term to dampen the oscillation
3. Increase the integral term to slightly shift the car

From this I was able to tune the speed controller and once I could see it doing well, I went back and adjusted the values for the steering controller, basing what I needed to adjust on my observations. For example, I could see that the current proportional value was over steering in respect to the higher speed, so tuned this parameter down until I did not observe over steer.