PID Control

Build a PID Controller

The goal of this project is to build a PID controller that drives the vehicle around the lake racetrack (already seen in Project 4: Behavioral Cloning).

The simulator provides the cross-track error (CTE) in order to compute the appropriate steering angle.

The goal of this project is the following: drive around the track safely and as fast as possible.

Rubric Points

Here I will consider the rubric points individually and describe how I addressed each point in my implementation.

Compilation

The code compiles correctly.
Code compiles without errors following the project instructions:

1. Make a build directory: `mkdir build && cd build`
2. Compile: `cmake .. && make`
3. Run it: `./path_planning`.

Implementation

1. The PID procedure follows what was taught in the lessons.

During the implementation of the PID procedure I used the methods that were discussed in the classes when determining the PID errors and the total error, too.

Reflection

1. Describe the effect each of the P, I, D components had in your implementation.

In my implementation I used the following understanding of the P, I, D components:

P: this is the proportional term, represents the present status of the system. It is also used to set the magnitude of the control.

In the simulator controlling the vehicle with the P term only resulted that for small K_p values (small output value) the vehicle was not able to follow the road. For larger values the vehicle could follow the road but started oscillating. Therefore, when setting the final values (refer to the next section for the method), the increment step of K_p was 0.1.

I: this is the integral term that represents the “past” of the system (past errors). It is used to make the process variable converge faster to the set point, but it also introduces instability (oscillation) in the system, therefore the value must be selected carefully.

In the simulator controlling the vehicle with the I term only is very funny, as the accumulation of errors results that the vehicle drives in a circle (until it pops up onto a ledge.) The increment step of the K_i parameter was therefore set to 0.001.

D: this is the derivative term, represents prediction to the future. This term is used to damp oscillation of the process value.

In the simulator the D term can drive the car on the straight section of the road (a large K_d value must be set), but it does sharp “corrections” to follow the road and cannot handle large turns. The increment step of K_d was set to 1.0.

It is interesting to note, that the car can be driven around the track a full lap using PD control, only. Nevertheless I used the I term, too, to improve keeping on track in the curves.

2. Describe how the final hyperparameters were chosen.

PID controllers have a long history in controlling systems and there are many methods used to tune the parameters.

I attempted to use the Parameter Twiddle algorithm presented in the class, but unfortunately my implementation wasn’t good enough to result usable parameters, so I had to “fall back” to another method we used at the university.

This is a manual tuning method that is based on the Ziegler-Nichols tuning (https://en.wikipedia.org/wiki/Ziegler%E2%80%93Nichols_method), but avoids having to determine the oscillation period and is a little bit simpler, too.

The steps are the following:

1. set all parameters to 0
2. increase K_p until the process value starts oscillating
3. increase K_d until the oscillation is damped
4. repeat steps 2 & 3 until the oscillation cannot be damped fully (and use the last stable values, of course)
5. increase K_i until either the system converges fast enough, or the amount of acceptable oscillation is reached
6. if the system cannot tolerate more oscillation but the process variable doesn't converge fast enough then a more sophisticated setting method is needed (process model needs to be established or refined)

Luckily, in this case this simple manual tuning worked. These are the final parameters:

$K_p = 0.2$, $K_i = 0.001$, $K_d = 7.0$

Simulation

The vehicle must successfully drive a lap around the track.

The vehicle is able to drive around the track safely. Although it does not drive fast and the drive in the curves is not smooth.