PID Controller

Andreas Gotterba

1. Edited Files

agotterba_pid_writeup.pdf	**This Document**
src/	
main.cpp	**main program**
PID.h	**basic PID controller header file**
PID.cpp	**basic PID controller implementation**
TWIDDLE.h	**twiddle header file**
TWIDDLE.cpp	**twiddle implementation**
Videos	
50 mph manual tuning:	**https://youtu.be/DCruNl8ALF8**
50 mph lap after twiddle:	**https://youtu.be/zKlGrUb3HNE**
50 mph curves:	**https://youtu.be/3UwPMevhsM8**
kp increased:	**https://youtu.be/gB4pnwOSRjc**
kp decreased:	**https://youtu.be/26EmOhDT2B8**
kd increased:	**https://youtu.be/3UwPMevhsM8**
kd decreased:	**https://youtu.be/ui_lbKMgp_w**
ki increased:	**https://youtu.be/bK8UP2Zi8Po**
ki decreased:	**https://youtu.be/RaRZzWEqtjw**
15 mph manual tuning:	**https://youtu.be/o5mbsOFGstw**
15 mph twiddle result:	**https://youtu.be/v7_jAliVolw**

2. Parameters' Effects

I'll discuss the effects of setting the parameters too high or too low, with '50mph curves' as the reference. Note that I tolerate oscillations over the bridge, so long as the behavior through the most difficult turns is good.

2.1 P

The workhorse parameter, P is the main means of reducing the error. It sets the steering angle proportionally to the cross track error (cte). On its own, this means that the car oscillates around the center line, as nothing tells it turn back towards the center line until the car has already crossed it. As can be seen in the kp_increased video, setting it too high increases the car's oscillation Setting it too low means the car is more stable over the bridge, but isn't as quick to correct when it encounters the turns, at one point hitting the ledge.

2.2 I

I, the integral term, adds to the steering angle proportionally to the integral of cte. The lectures describe it as correcting a bias in the system, but that is with regard to a straight path. I rely on it most to help through turns. If driving a circular track, a PD controller will find a steady state with a radius larger than the circle, as it needs to be off the center line for the steering to turn at all (D is doing nothing, since the derivative is zero). The I term lets it get much closer to the centerline, without setting P too large. Setting I too large should increase the oscillations, but that's not what my video shows. It oscillates with large steering corrections, but does a good job of staying in the center of the road. This could be said to perform better, but I don't like how jerky it is (and ultimately, I chose my parameters with twiddle, so I didn't make a qualitative assessment). Setting it too low reduces stability, and makes the car turn far too late (it was hard for me to find a lower setting that even completed a lap). As my I term is much larger than my P term, I expect that for me, I is doing much of the work that P would normally do.

2.3 D

The D parameter reduces oscillations, by reducing the steering angle when the derivative of cte (implemented as simply current_cte – previous cte) is falling. This allows the car to turn back towards the center line as it approaches it, meaning it won't overshoot so far on the other side. However, setting it too high can make the car less stable (I suspect because there is delay from when the new value is calculated to when it is applied, to strong an input can be incorrect by the time it makes it to the car). Setting it too low decreases stability, as both the P and I terms encourage the car to oscillate around the center line. While it looks good with low D through the turns, its oscillations on the bridge are increased, and it touches the dirt on the first turn (as D also helps increase the turning radius when cte is increasing in magnitude). I would have liked a more dramatic example, but there isn't much margin between visible degradation and failing to complete the course.

3. Parameter Choice

While I ultimately used twiddle, I had to go through a process to get to the point where twiddle could be used.

The PID controller for the throttle used manual tuning- I set it so that the speed is close to target, and the car doesn't brake too often (from overshooting).

3.1 Initial Manual Tuning

I started by having the car drive at 15mph, and manually adjusted the parameters until the car made it to the bridge. I was able to do this with only the P term, with a setting of 0.12.

3.2 Ziegler-Nichols

Once on the straight bridge section, I could measure the period of the oscillations, and apply the Ziegler-Nichols approximation, which is discussed in this paper by George Gillard (linked on the forums). I approximated time by counting time-steps from when cte swung from positive to negative and back (though I later added code to work in units of absolute time). From this calculation, I found that the I term should be much larger than I expected- an order of magnitude larger than even the P term. While this didn't work- the car didn't complete a lap- it's what got me expecting the I term to be large.

3.3 Twiddle

My master plan was to use twiddle, and once the maximum cte was small enough, and the sum of cte^4 was no longer improving, gradually increase the speed. It had become clear that different speeds required different settings (the use of a PID controller for the throttle was essential). I created TWIDDLE.h and TWIDDLE.cpp, so that this code would not be mixed with the simple PID controller (in PID.h and PID.cpp). Its implementation is more complicated than it is in the lab's lectures, since the simulator runs independently. We can't tell it to run a lap while the twiddle code waits for a result. Thus, I track the state with step_count to time approximately one lap, and the counters param_ring and posneg_ring to remember which parameters have been modified, and evaluate their results.

In short, it didn't work as I had planned. I got decent results at 15 mph and it would advance to 16mph, but never behaved well there. And once twiddle had finished optimizing 15mph, the D terms it found were negative. I tried lots of things to improve it:

1. The Gillard paper talked about integral unwinding- I tried setting a maximum value, and increasing the rate that its magnitude fell when the integral and the cte had opposite signs. I had briefly tried

setting the integral to 0 when the car crossed the center line, but found the car needed to over correct a little while in a turn, or else it would quickly fall back to the outside.

2. I tried adding an additional PID controller that takes cte^3 as input, and add the result of this controller and the main PID controller. My thinking was that I wanted it to do almost nothing when cte is small, but when cte is large, give the correction a stronger response. I've left this in, but the parameters twiddle found for it are too small to be useful.

3. Since I don't car about small cte, but really want to prevent large cte, I sum cte over a lap as cte^4 instead of cte^2.

Most people on the forums discuss running at 50mph, so I went back to manual tuning, found values that got around the track (taking inspiration from the large integral term from Ziegler-Nichols), and then applied twiddle again. The values it found didn't change much from my manual tuning.

I think that in this context, twiddle has a few problems:

1. Noise: there's quite a bit of variation from lap to lap with the same parameter settings. I'd find improved parameters with a summed error of 1716, only to get 2064 when I ran with the same settings at the start of the next epoch. With so much noise, twiddle won't be able to make small improvements- the improvement must be large to overcome the noise.

2. Related to noise, twiddle can get stuck in a local minimum of summed cte, and the noise prevents it from getting out. I highly suspect that the cte^3 PID controller could be more useful- a limitation of PID controllers that I saw cited is that they're linear, but supplying a non-linear input is a way around this, similar to what we did with SVMs. But the normal PID controller's P term likely needs to decrease before its P term can increase. If it wasn't for noise, this should happen naturally once the increments get small enough, but it never did. Some people on the forums recommend restarting the simulation every time settings change, but I had problems with the simulator and my code losing their connection when this happened, so I turned it off.

4. Performance

So after all that, I am pleased with my car's performance.   It touches the red and white stripes in a few places, but so does Louis Hamilton. Also, there are oscillations when it first starts as it needs to get up to speed in order for the parameters to be correct. There's certainly room for improvement- I'm disappointed in my twiddle result, and think I would have better performance if I'd spent that time on manual tuning.