Write up

1. How to meet the rubric of PID control project.
2. By control the cte that was sent back from simulator, to get the vehicle run smoothly.
3. Control the throttle and speed, according the cte.
4. Road map of get the target
5. Manual tune-up the PID parameter in main.cpp, and just control the steer\_value according cte.
   * No control

Set PID parameters as 0/0/0, the vehicle was driven out of the line as we can image in about 3 seconds.



Figure 1 without control

* + P control

Firstly, I would like to try out the P control. Which value should I set as P?

I don’t know…I found in the previous tutorial that Sebastian showed, the P parameter was set as 0.1. Um mm, let me try this value also. ☺

This time the vehicle can be driven in the lane smoothly for about 20 seconds (3 seconds when without control), but at the 21th second the vehicle begins to shake largely and at the 23th second it was driven out of lane. Which can be seen in figure 2 and figure 3.

Now I want to change the P parameter, even though I know P control leaves some error that can’t be cut down to zero from Sebastian’s tutorial.

I list some of them in the following table.

|  |  |  |
| --- | --- | --- |
| P value | Vehicle  within lane time | remarks |
| 0.1 | 20 | Initial value |
| 0.2 | 18 | Random try |
| 0.05 | 24 | Better than the Random try |
| 0.025 | 23 | Somehow the same as 0.05,  but can come back sometimes  when come out of the lane |
| 0.0375 | 20 | Bad than 0.025 |

After trying out different P parameter, I came to thing about why the vehicle was driven like that? I change the n parameter from 100 to 1000 which means extend the step we can see and P parameter in the python version P control code. Finally, I got the figure 4 with n=100 and p=0.1,

Figure 5 with n=1000 and p=0.1,figure 6 with n=1000 and p=0.025.

As we can see, with fixed P parameter only, the controller will oscillation at last, the different with selected P parameter with 0.025 will last longer than 0.1.

Until know, I think I found one method to manual change the parameter and how to foresee if the parameter will work or not. ☺



Figure 2 with P=0.1 the vehicle was driven with in the lane



Figure 3 with P=0.1 the vehicle was driven out of lane at the 23th second

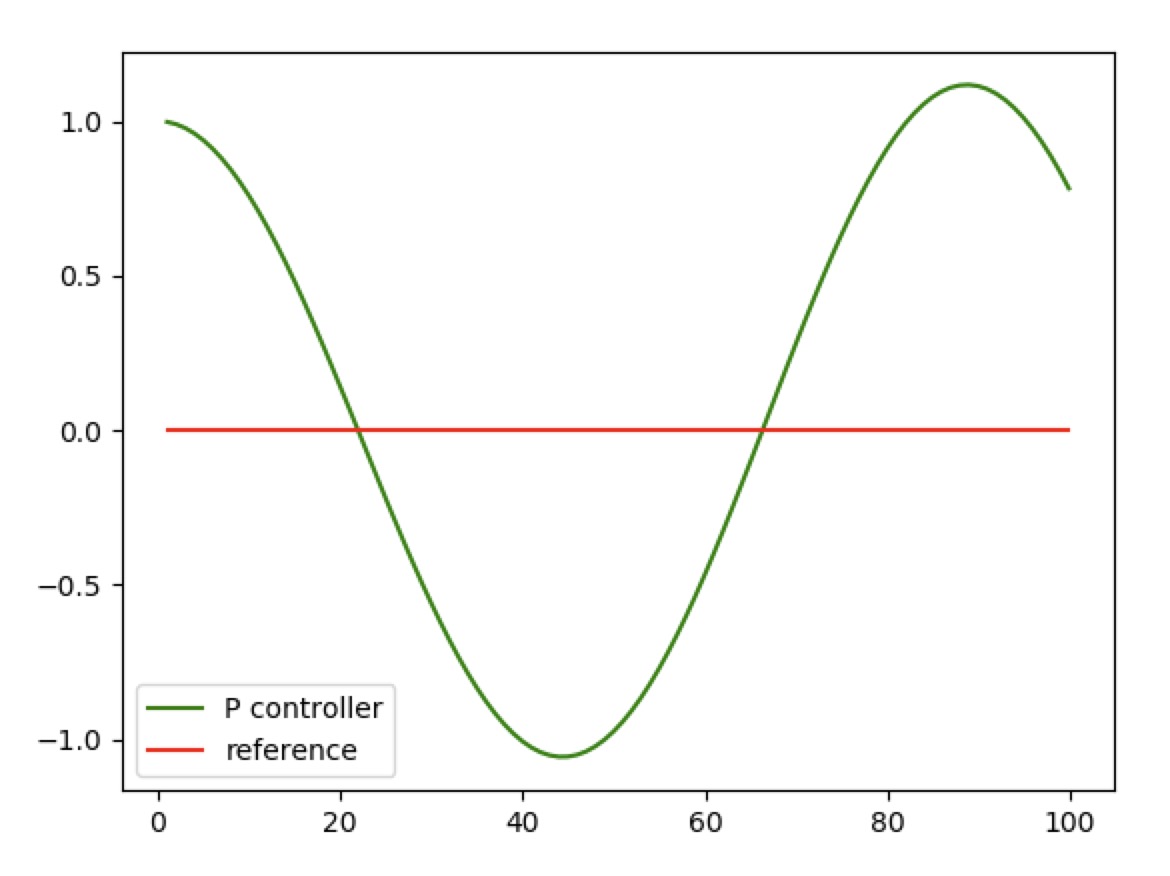


Figure 4 P controller with n=100 and P=0.1

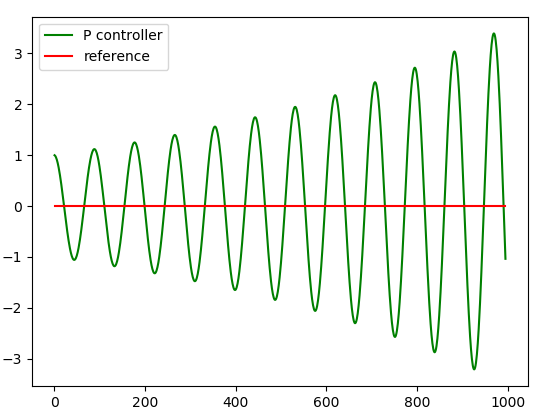


Figure 5 P controller with n=1000 and P=0.1



Figure 6 P controller with n=1000 and P=0.025

* + PD control

As that was tried in P control, this time I would like to try tune-up parameters in python version firstly. I set P=0.2,D=3.0 and n=1000. As showed in picture 7, the controller seems stable. I can’t wait to try this parameter in C++ version. When the vehicle runs one round and second rounds, I got so excited. But, I can’t believe I have finished this project because if it were, it would be kind of easy. Wow, as showed in picture 8, the steer angle was -1.58 which doesn’t meet the rubric that should between -1 to 1.

What’s wrong, what can I do next step?

I zoom in figure 7, as showed in the figure 9, there is cycle oscillation even it is very small. I don’t know the relationship between this cycle oscillation value and the steer angle value, but I guess it should be the root cause that the steering larger than 1 degree. BTW, with the PD controller, the vehicle can run several rounds, so I don’t believe this system have large bias that need I control. Now, I decide to try tune-up the PD parameter to meet one round and steer angle is also with in 1 degree. If I can’t manually find the suitable PD parameter, I would limit the steer angle amplitude to meet the rubric. I change P parameter from 0.2 to 0.3 and got a little smaller error as showed in picture 10. But in the simulator, I could find obvious difference compare to the previous P parameter and these parameters cause a little large shake at the beginning of the vehicle was driven.

I think, I can’t control steering value with in 1 degree. This made me in deep thinking why 1 degree. I go through the rubric again, find nothing that ask us to limit with in 1 degree. Um, mmm…I am sure it is not my illusion…

Finally, I found one sentence in the main.cpp

“remember the steering value is [-1, 1]”

Alright, I think the value’s unit should be radian, which means [-57.3,57.3] degree. This misunderstanding is not bad thing, because it made me to think how to make the vehicle driven with less shake.

Back to the previous question, is it true that this project is so easy. Um, mmm, I think it is just because I followed the parameter in Sebastian tutorial. I set P as 0.2 and D as 3. How should I get the parameter, if I didn’t see them in the tutorial? This made me though about the sentence in the tutorial by Sebastian:” Twiddle is my favorite algorithm that I have used in my entire life”

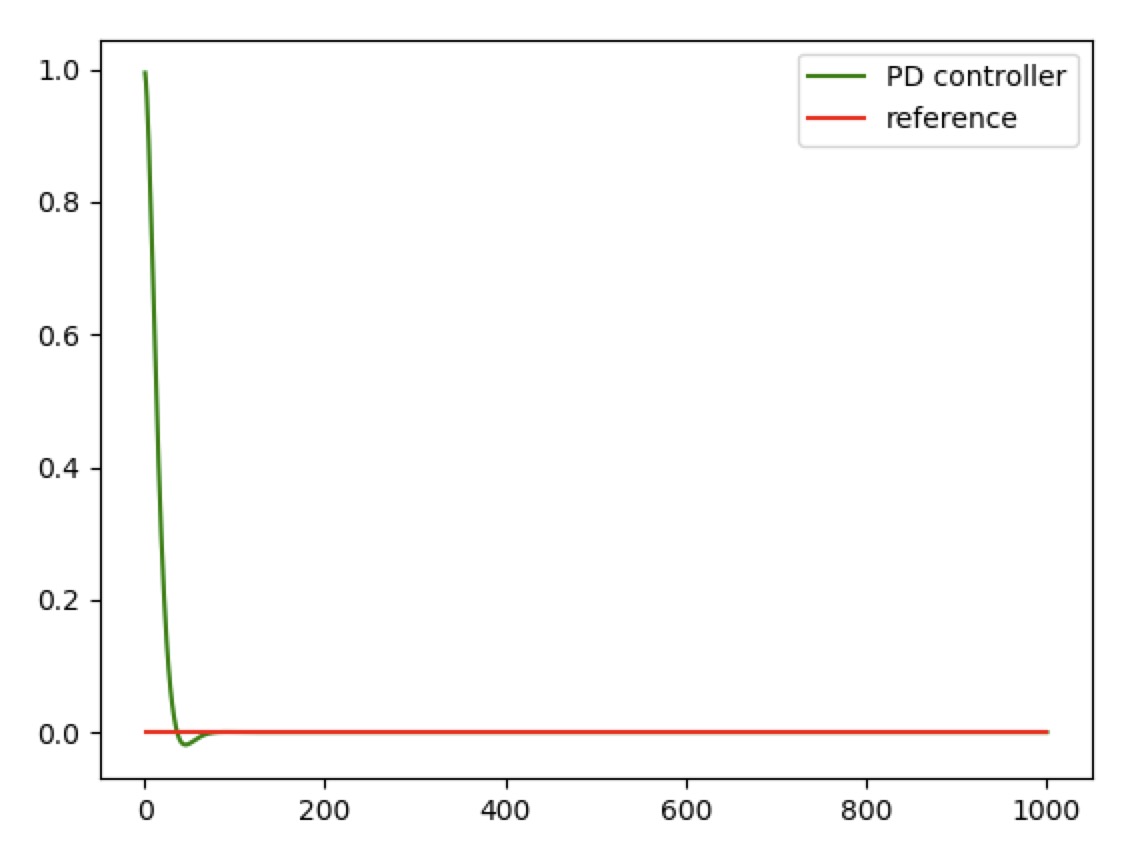


Figure 7 PD Controller with P=0.2, D=3.0 and n=1000



Figure 8 Vehicle runs several rounds with PD Controller with P=0.2, D=3.0 and n=1000

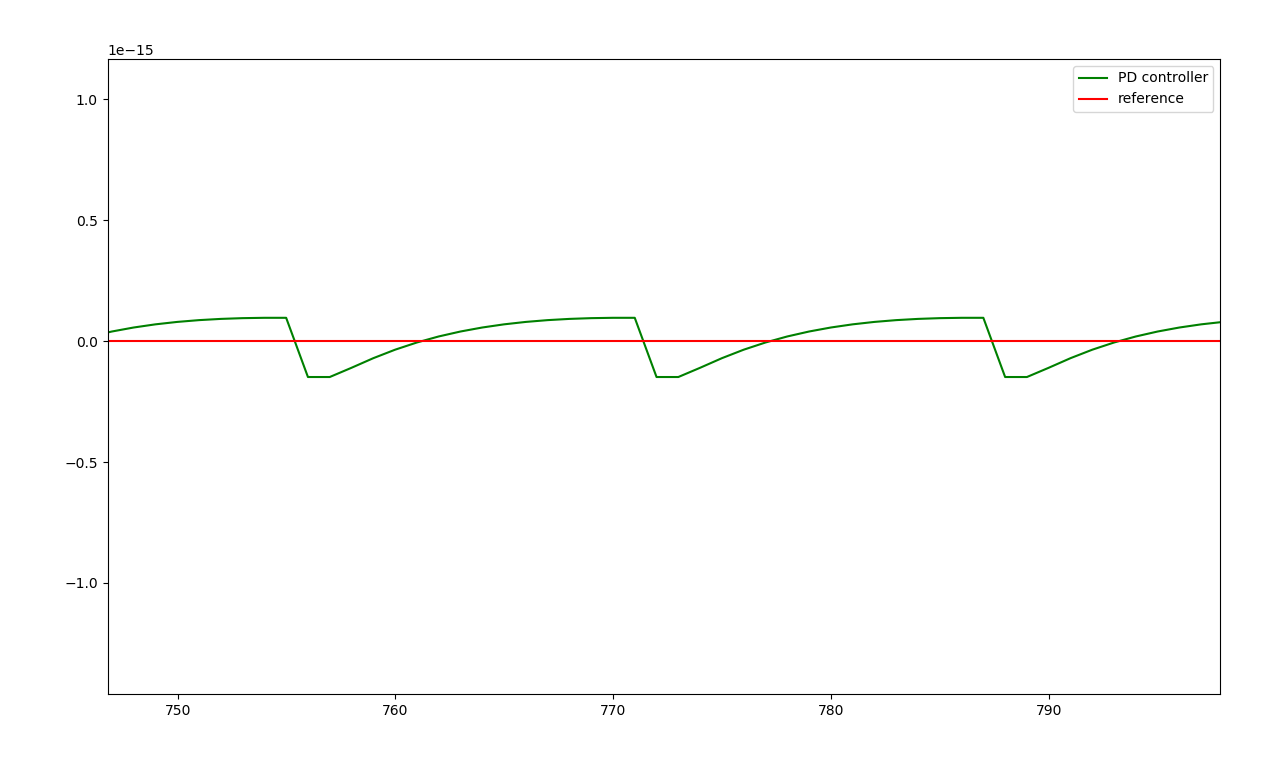


Figure 9 the PD Controller error with P=0.2, D=3.0 and n=1000

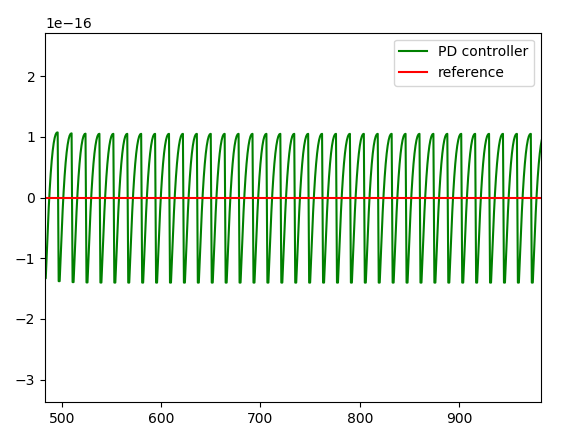


Figure 10 with P=0.3, D=3.0 and n=1000

* + PID control

I will leave this part white, because the successful result of PD controller.

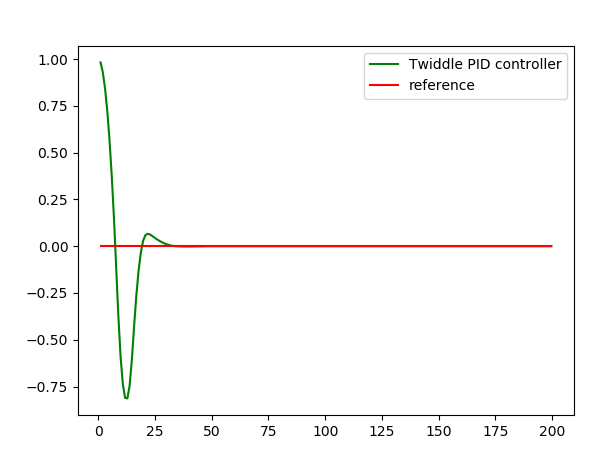
* + other

1. Using twiddle method to get the suitable PID parameter and implement in PID class

I used the python version of “Twiddle PID controller” which base on Sebastian’s tutorial. And got the PID parameters as P:2.9 I:0.49 D:10.32. I tried these parameters in the C++ version, this time the vehicle shakes largely.

Um mmm, what happened.

After several minutes thinking and checking the code, also recall my friends who works in control field. One word came into my mind again, it is control target. Yes, the python version code has different control target with c++ version. In details, the python version’s control target is implemented as *move* function in Robot class.



1. Deep in thought

Control target can never be exactly described by mathematic model. It can only be as exactly as possible. Base on this though, I think it is enough of this trying of tune-up PID parameters.

Actually, it seems I got the parameters by chance. ☺