PID controller is one of the most common way to stabilize a closed loop system. A system is closed loop if a portion of output is added back to input of system. This extra input is known "feedback signal" which helps to stabilize output. This feedback is added in 3 variations. One is feedback which is proportional to error signal, 2nd is integral of error and 3rd is derivative of error term.

1.Proportional tuning involves correcting a target proportional to the difference. In this feedback is added in Proportional to current system error. So a system oscillates around a target value but is never approaches stable final value.

2. The integral part help to remove the oscillations from Proportional tuning, it is due to control effects of historic commutative error begin added which eliminates error. However, when it reaches target values this error term also reaches zero.

3. The last part is derivative term, which tries to minimize overshoot of system by slowing down it responses to error. This does not impact target values for system.

Below table captures impact of varying these parameters in a system

Parameter	Rise time	Overshoot	Settling time	Steady-state error	Stability
P	Decrease	Increase	Small change	Decrease	Degrade
I	Decrease	Increase	Increase	Eliminate	Degrade
D	Minor change	Decrease	Decrease	No effect in theory	Improve if D is small

There are many ways to tune a PID system like Manual tuning, Ziegler-Nichols , Cohen-Coon and many others. For this project I tried to implement Twiddle also as taught in class. It is as per below algo.

```
p = [0, 0, 0]
dp = [1, 1, 1]
best_err = A(p)
threshold = 0.001
while sum(dp) > threshold:
    for i in range(len(p)):
        p[i] += dp[i]
        err = A(p)
        if err < best_err:
            best_err = err
            dp[i] *= 1.1
        else:
            p[i] -= 2*dp[i]
            err = A(p)
            if err < best_err:
                best_err = err
                dp[i] *= 1.05
            else
                p[i] += dp[i]
                dp[i] *= 0.9
```

I tried to implement initially but after it was taking quite some time to pump out optimized params due to which car moved to new location and those values were not correct for that location. So I tried to do some manual tuning for this project. For that I pass values for P, I and D params while initiating my program. With those values I was able to run 3 rounds around the track as attached in video.

```
gasati@GASATI-LT1:~/sdc/CarND-PID-Control-Project/build$ ./pid 0.09 0.001 0.55
     0.09   0.001   0.55
Listening to port 4567
Connected!!!
```