PID Controller Project SDC

Overview

“PID Controller SDC” , this project involves to code a PID system for controlling(atleast) the control variable “steering angle” based on the input “Cross Track Error” which basically is the amount of distance between the center of the lane and the centroid of the box model of car in simulation.

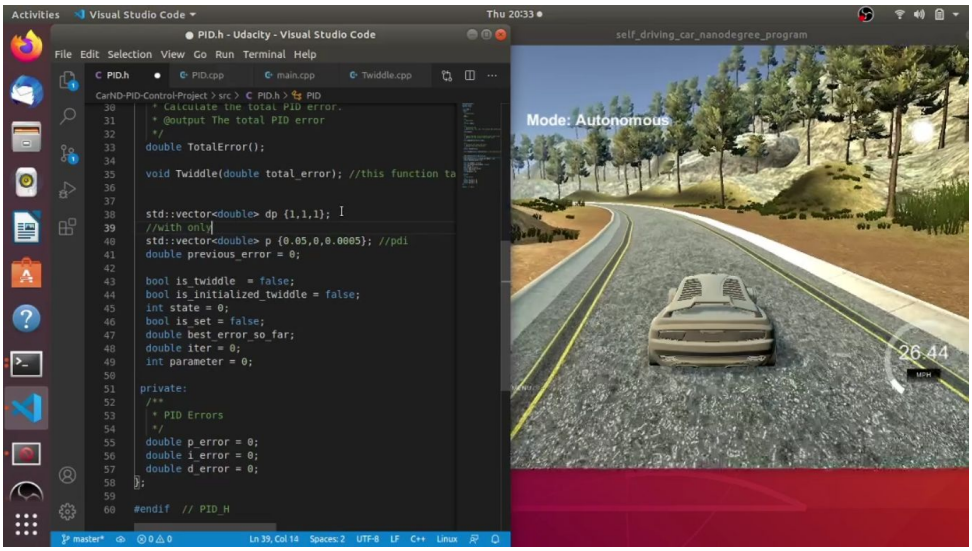
Initial values of PID parameters are set manually by individually varying parameters, then used twiddle optimization algorithm for about 5 iterations each with a step size of 300 to optimize the manually obtained parameters.

Goals

1. To code PID algorithms from scratch in CPP.
2. To code Twiddle algorithm .
3. To make a car in a simulation maneuver around the track using a PID controller.

Brief

- “PID controller” stands for proportional , Integral,derivative controller. Proportional part takes the cross track error and changes the control variable ,here steering angle, proportional to the amount of cross track error. This is good for small value of errors , but for large values of eros system tend to overshoot , this makes the object , here car , go off the track or desired value it intended to achieve. Following video shows the output of controller with only P parameter set , to value 0.05.



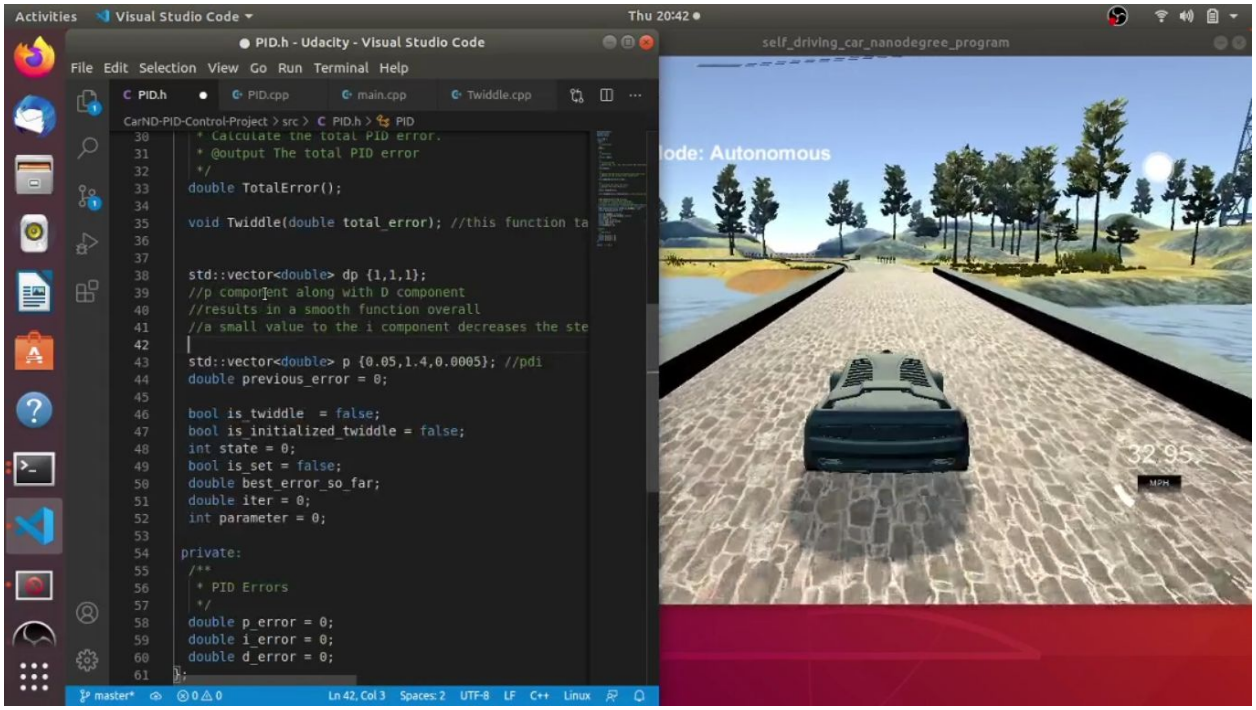
Link to Video , showing output with only P parameter set.

- When only the p-controller is set, the system oscillates vigorously about the desired value i.e., center of the lane . In order to kill this behaviour and make the response die out as we progress we use a Derivative part of the controller . This works like a feedback , gives the system a memory about the performance of the system in previous timestep and thereby controls present behaviour. We approximate this derivative by a first order difference equation with assumption of time gap 1 second. Following video shows the output response of the system when both P parameter and D parameter are set , and also I have set a very small value to the I parameter , this decreases the steady state error , this accumulates overall previous error (integration) and changes the output response .

  P parameter : 0.05

  D parameter : 1.4

  I parameter : 0.0005



Link to the video showing the output response of over all PID controller.

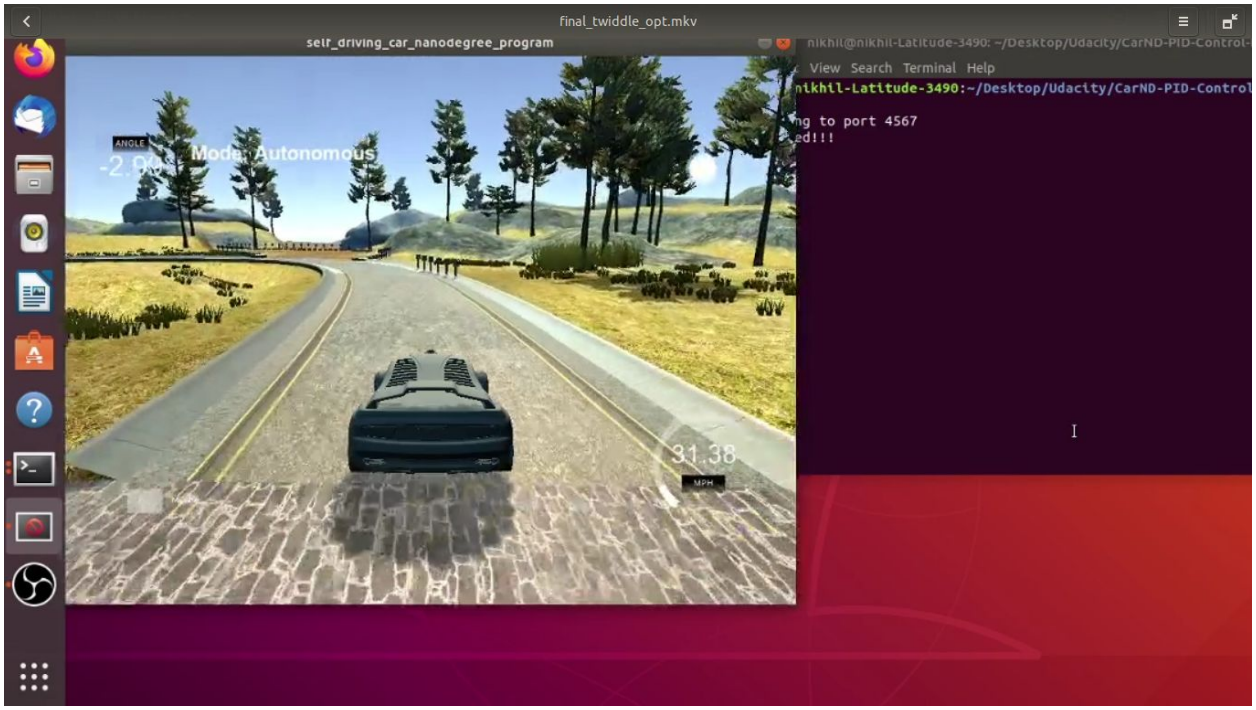
- Now we got a good set of PID parameter values , but in order to still refine the output we could still refine the parameters i.e., optimizing the parameters , for this kind of task we generally use optimization algorithms like gradient descent , Twiddle algorithm , Stochastic Gradient descent etc . For them to perform better we generally set a good initial value to the parameters , this is done to achieve optimized values fastly that is with good initialization we restrict search space. For this project I have a twiddle algorithm for optimization tasks . I have run twiddle algorithm for 5 iterations and obtained optimized PID parameters are

  P parameter : 0.084

  D parameter : 2.4

  I Parameter : 0.0005

(NOTE: even though i have run the twiddle for 5 iteration , i have achieved above values in 3 iterations , remaining iteration only made things oscillatory).



Link to video showing output of PID controller with Twiddle optimized parameters.