PID Controller Project

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

- Proportional:
  The proportional term produces an output value that is proportional to the current error value. The proportional response can be adjusted by multiplying the error by a constant Kp, called the proportional gain constant. The proportional term is given by:
  P = Kp * crosstrack_error
  So the proportional part of PID controller tries catch up the reference line and tries to minimize the error, but because of it car overshoots, and wobbles a lot. So I have kept Kp=0.2, which is low so that it doesn't overshoots much.
- Integral:
  The integral in a PID controller is the sum of the instantaneous error over time and gives the accumulated offset that should have been corrected previously. The accumulated error is then multiplied by the integral gain (Ki) and added to the controller output.
  integral_error = sum(past-crosstrack errors)
  I = Ki * integral_error
  So the integral part of PID controller tries to eliminate any possible bias in steering. Since, no bias was present in the simulator, so I have kept Ki=0.004 which is very small.
- Derivative:
  The derivative of the process error is calculated by determining the slope of the error over time and multiplying this rate of change by the derivative gain Kd. The magnitude of the contribution of the derivative term to the overall control action is termed the derivative gain, Kd.
  derivative_error = current_cte – previous_cte
  D = Kd * derivative_error
  So the derivative part of PID controller tries to counter the overshoot due to proportional by smoothening the approach to the reference line.

Q. Describe how the final hyperparameters were chosen.

I followed manual tuning of hyperparameters. I tested various sets of values of Kp, Ki, Kd and then found out that the values Kp=0.2, Ki=0.004, Kd=3.29. I twiddled manually, as in I changed a single parameter then checked the effect on error, and then changed the other parameters one by one. First I checked if the car is running straight with 0 as parameters, and then incremented the proportional gain and the car started to follow the road but started wobbling a lot because of overshooting. Because the car had no bias in the simulator which I confirmed when I put 0 as the parameters, the car was running straight, so I kept minimal value of integral gain. Then I added deferential to overcome the effect of overshooting due to which the wobbliness reduced.