Student describes the effect of the P, I, D component of the PID algorithm in their implementation. Is it what you expected?

P is proportional. It changes the controller output in proportion to the error. This means the control action will be bigger if the error is bigger. For this project, if CTE is larger, the proportion part of the control means more steering wheel changes to make it driving correctly. If the controller gain is set too high the control loop will begin oscillating and become unstable, which does show in simulator when Kp is large. If the controller gain is set too low, it will not respond adequately to disturbances or set point changes. In this case, the car will not handle error correctly and drive outside the road.

I is integral. The integral control will continuously increment or decrement the controller's output to reduce the error. Given enough time, integral action will drive the controller output far enough to reduce the error over time. If Ki is too large, the controller will be sluggish. But if is too small, it will lead to oscillate and become unstable. This project seems does not have large system error, therefore Ki is small here.

D is Derivative. The derivative control produces an output based on the rate of change of the error. Derivative mode is sometimes called Rate. The derivative mode produces more control action if the error changes at a faster rate. If there is no change in the error, the derivative action is zero. A derivative time setting of zero effectively turns off this control. If the Kd is set too large, oscillations will occur and the control loop will run unstable.

Describe how the final hyperparameters were chosen.?

Basically, I manually tuned all three gains.

The final value I chose is (0.12, 0.00001, 3.0).

The step is as follows:

1. Firstly initialize all three values to zero. In this case the car will drive out of the road immediately
2. Then tune the Kp value, to make it able to follow the road route, and not oscillating too much.
3. After Kp, add some Kd value to make the control more stable from the oscillating. At first my value was too large, resulting is overshooting. Then I fine tuned the value to make the car drive in the designed route smoothly.
4. Finally add some Ki, to be against some offset. It does not show a big system error in this project. Therefore it is a relatively small value.
5. The car is able to drive in a circle successfully.