reflection

in the project attached, I used a modified version of PID algorithm to determine the PID coefficients.

the PID model is compound of 3 components:

1. the P component which "pull" the steering wheel relative to the cross track error (cte).

2. the I component which responsible for any compensation needed for any distortion.

3. the D component which his goal is to ease the turn rate (cause usually by the P component) relative to the progress made in time by the model.

in order to find the right coefficient to those components, I used the twiddle algorithm with little differences.

here are the two main differences:

1. I used square cte instead of just cte to calculate the mean error. this change allows me to penalize the model for bigger distances.

2. if the error was bigger than the best error, instead of moving on to the next iteration I measured if the error got any better since the last round. but only if the other two components of the PID coefficients were not updated in the last round. this way I managed to avoid local minima that I encounter.

the results were interesting. instead of getting results where the P component is the largest, and the I and D components are smaller, I got this results: { 2.77054, 0.0796149, 27.532 }. note the huge value of the D component.

at first, I thought that the model will never converge, but in the end, the D component got so big, that instead of smooth driving, the car is zigzagging on the road, consistently trying to correct his small mistakes. this could be the result of the square cte method that I used.

but with all of the unsmooth behavior, using those coefficients the car is driving in the middle of the road, without falling out.