Philippe Rougier

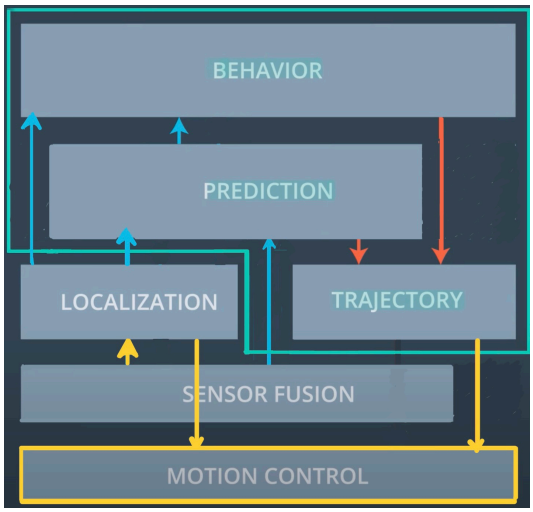
PID controller

Goal:

Design a software to continuously adjust the motion of an autonomous vehicle so it follows its planned trajectory.

Autonomous Driving data flow & Motion Control (PID controller) :

Autonomous Vehicle Driving Data flow :



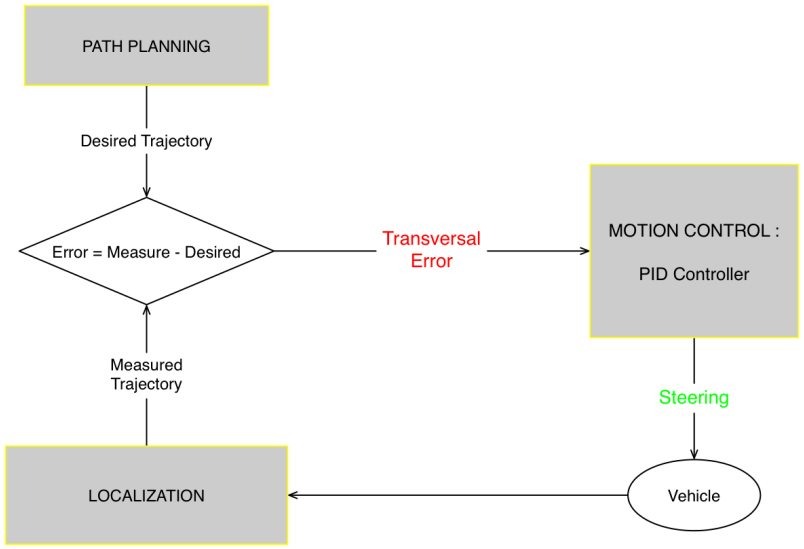
We addressed Sensor Fusion, Localization and Path Planning (Prediction, Behavior, Trajectory) in previous projects .

The PID controller, as part of the Motion Control module, is at the end of the pipeline. For this project, we'll only focus on the transversal position of the vehicle on the lane. Speed and longitudinal coordinates could also be monitored by a PID controller.

PID controller :

The role of the PID controller is to apply corrective steering as soon as the vehicle deviates from its planned trajectory.

Planned Path and actual Vehicle Localization are continuously compared. The PID controller respond to any difference (Cross Track Error) by immediately outputting the remediating steering angle.



Due to the sequential aspect of the Motion Control process (discretization of a linear process) the PID controller addresses the Cross Track Error (transversal error) from 3 perspective for **each measurement step** :

- **Proportional Error**: proportional to the CTE at each measurement step
- **Derivative Error** : proportional to the CTE **variation** between each step
- **Integral Error** : proportional to the **sum** of CTE along the path

As it is easy to understand the Proportional Error correction (the farther away the car drives from its intended trajectory the more corrective steering to apply), the Derivative Error is less intuitive.

It can be interpreted as a way to measure how fast the CTE is corrected (by the Proportional Error correction) and to address this latency.

As for Integral Error, it has more to do with correcting chronic differences between the intended steering and the vehicle response to it. It will address a vehicle continuously drifting to one side of the road (faulty wheels alignment, uneven tires pressure, constant side wind …)

The scope of the project is limited to :

- calculating theses errors
- finding the coefficients to calculate the corresponding overall corrective steering

Technical Set Up Overview:

Drive Simulator :

A provided simulator reproduces the virtual of an autonomous vehicle in motion and its surrounding environment:

- the **CTE feed** is directly provided to us in lieu of a localization feed
- the PID module mission is to calculate the ad hoc steering (within a [-1,+1] span) and submit it to the simulator .
- this simulator updates the vehicle position accordingly to the PID output steering

Data format and programing environment :

The path planning software is implemented in C++ .

Implementation and fine tuning :

The implementation of the PID Controller is straight forward : the 3 types of errors are calculated for each step of the motion and summed :

total error = Kp . CTE + Kd . D_e + Ki . D_i

CTE (cross transversal error) = (position measured - position measured) from simulator

D_e (Differential error) = CTE - previous step (CTE)

I-e (Integrative error) = sum of all CTE till this step

Finding the right coefficients (Kp, Kd, Ki) is a bit challenging as the car can spin, oscillate and go off road when the coefficient have not be tuned adequately.

Twiddle, a probing algorithm , can help to narrow down on the right coefficient for this specific car.

Twiddle, in a nutshell , increment or decrement the value of each coefficients depending how these variation impacts the value of the CTE .

- a failing increment is followed by a decrement of the same value, if this attempt fails too then the value of the following increment of decrement attempt is reduced ((by a 0.9. factor)
- a successful increment (CTE decreases) is followed by a larger (by a 1.1. factor) one .

The Algorithm keeps looping through each coefficient till the increments are smaller then a a threshold (the error reaches a minimum).

Results :

We experimented with Twiddle (on one coefficient K_p) . Ideally twiddle should be implemented without affecting the driving . Probing the coefficients and calculating a projected CTE without affecting the steering value is a bit too involved for the scope of the project .
We also learned in this project that the initial coefficients and increments have to be within the range of the best coefficients to be tunable using Twiddle (other wise the car will prematurely drive off the road or the steering could flip right and left alternatively at each step) .

We conducted our trial and error in this order:

- we initialize Kp, Kd, Ki to zero
- we increased Kp till the car could drive properly the first 20 seconds
- we then proceeded the same way with Kd to fix the zig zagging
- then we tuned Ki so the car remained centered (critical to pass the sharp curves)

After many trial an errors our best (Kp, Ki, Kd) set of coefficients is (0.085, 0.001,1.5).
Movie:

Conclusion :

As Ki seems dependent on the vehicle condition (wheel alignment …) or environmental factors (side wind …) , it will be interesting to know how often these 3 coefficients are tuned.