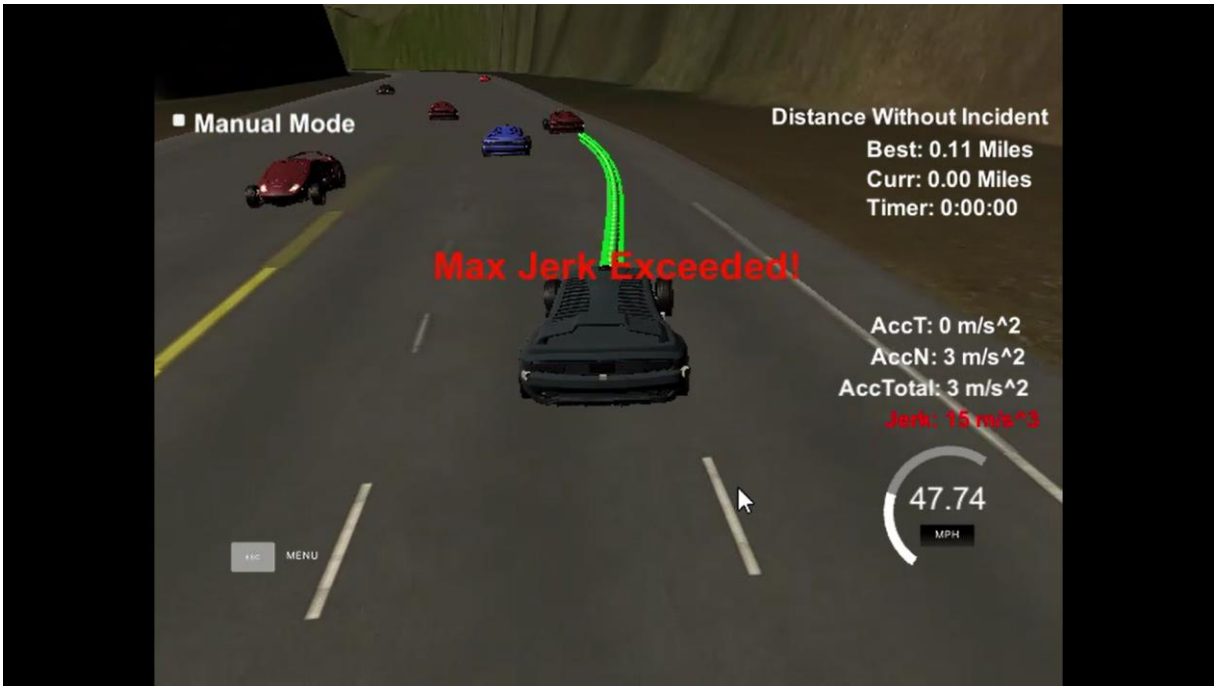
CarND-Path-Planning-Project



Introduction

The purpose of the Path Planning Project is to solve the problem of self-driving car in a virtual highway by smoothing the path and planning the driving experience behavior of the autonomous car. This leads us to consider the following requirements :

1. Smoothing the existing waypoints
2. Generating minimum jerk trajectories based on a minimal set of controls
3. Determining the best action based on model cost analysis
4. Transforming the generated trajectory from Frenets coordinates to cartensian ones

To do this, we neeed to follow the approach presented in the courses

Analysis and Design

During the desing phase, i apply some principles seen during the courses like classes (safety, feasability, comfort, ...). The Analysis of the project's given data leads to different logical components :
1. **Vehicle** : Responsible of keeping track of its current lane, its state, neighbor vehicles, front and back gap
2. **Lane** : Reponsible of providing information on the current lane like distance and neighboor lanes
3. **Behavior Planner** : Responsible of computing gap, cost for the ego vehicle and finaly decide of the lane changing
4. **Trajectory** : Responsible of velocity limitation, and data preparation for JMT computing
5. **Jerk Minimizing Trajectory** : Responsible of smoothing the trajectory by constraining the calculation process to the minimum jerk
6. **Path Transform** : Responsible of transforming the generated path in frenet coordinates to cartesian coordiantes
7. **Main** : Responsible of handling requests of way points from the simulator, calculates the ego vehicle environment, the ego vehicle trajectory by minimizing the jerk and sends back the converted the trajectory in cartesian coordinates to the simulator

Logical Architecture Description

Vehicle Logical Component:
Each vehicle encapsulates some attributes and methods to manage its own information provided by the simulator
Attributes :
1. **Identifier** : makes the vehicle unique among the set of vehicles on the scene. This attributes is assigned automatically except for the ego vehicle which is assigned manualy (id = 1000)
2. **Left Gap, Right Gap and Current Gap** : These attributes are of type "Gap". The Gap type component memorizes the distances from the ego vehicle to the one infront of it and to the one inback of it for each of the left, right and current lane of the egovehicle. it is used for gap calculation by the Planner to make decision on lane change
3. **S and D states** : These attributes keep track of the s state(p=s,v=speed, a=0n-- we consider the velocity is constant) and the d state (p=d,v=0,a=0). The s and d states are used in the trajectory calculation
4. Lane : it represents the current lane. The lane attributes pointes to the logical component : lane and therefore can exploit its contained information

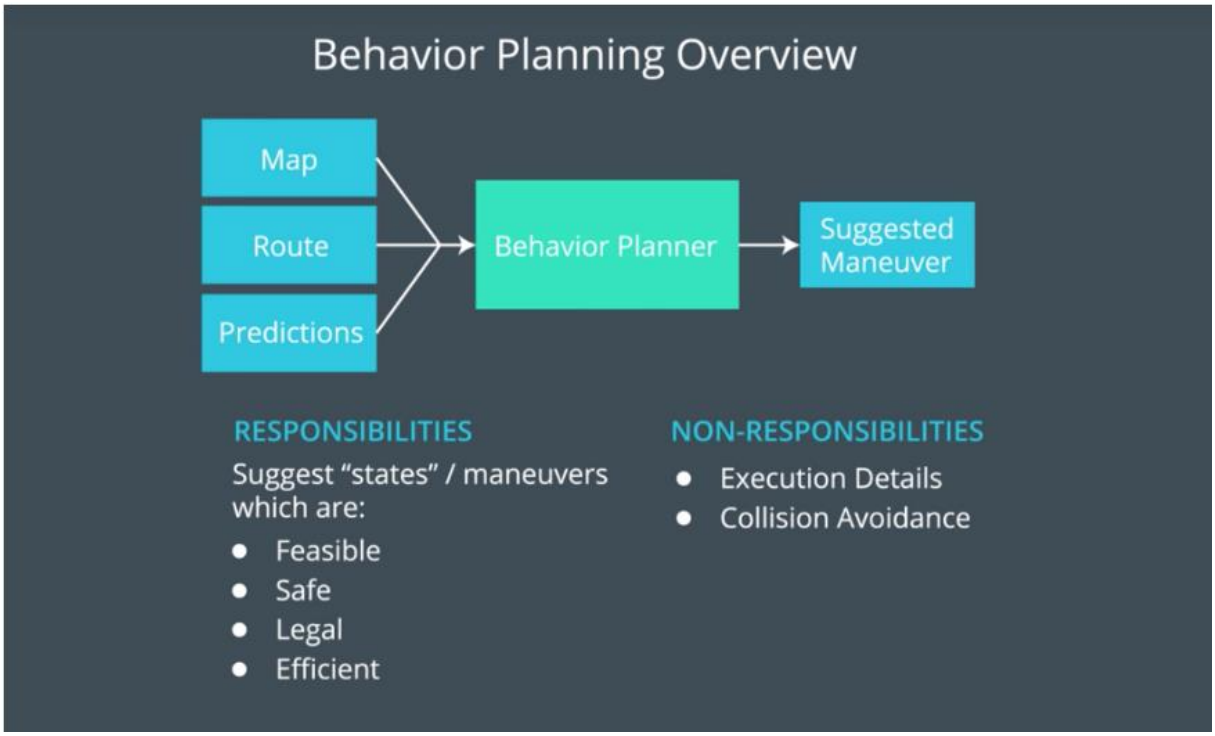
Methods :
1. **Update** : The update method save the start state and the current lane position and its neighbour lanes
2. **update_environment** : This method takes the neighbor vehicles built set using sensor fusion data provided by the simulator and determines the closest neighbors front, back vehicles' on the current lane and the neighbor lanes left, right. It compputes then the gap to the ego vehicle from each determined closest neighbor vehicle.
3. **compute_gap** : This method is used to compute the gap between the ego vehicle and one of the closest neighbor vehicle
4. **print** : This method is used to print out the different characteristic of the vehicle for debbuging purpose

Lane Logical Component:
The lane component is pointed by the vehicle component and stores by delegation the ego vehicle lane position and neighbour lanes
Attributes :
1. **type,left_type,right_type** : The type attributes stores the type of the current lane, the left_type stores the type of the left lane and the right_type stores the type of the right lane. Knowing that these attributes are of user defined type : LaneType

Methods :
1. **target_lane** : This method takes a lane position d in frenet coordinates and computes the target lane that will be the current lane lane and the right and left the lanes.
ex : if d = 2 ==> the target lane will be of TypeLane::Left, its left lane will be TypeLane::None and the right lane will be of type LaneType::MID (middle)

This method is invoked by the vehicle component

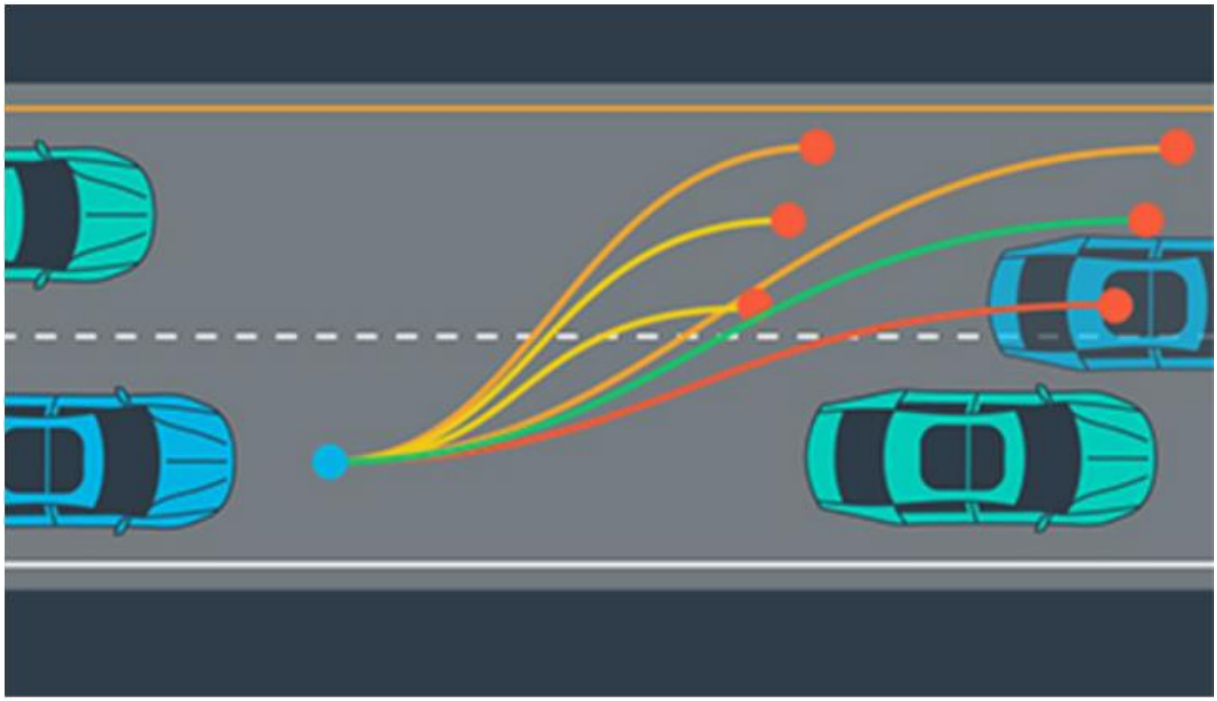
Behavior Planner Logical Component:
The Behavior Planner component is called by the main component make decision on lane change



Attributes :
1. **type** : The type attributes stores the type of the B Planner,this attributes is of type BehaviorType::KEEPLANE, TURNLEFT, TURNRIGHT
2. **cost** : The cost attribute stores the commputed cost related to each gap (front, back) and for each lane (current, left, right)

Methods :
1. **compute_cost** : This method computes the lane related cost which resulting value is between [0,1] according to the sigmoid function. The algorithm apply a penalty coefficient to the left and right costs during the calculation to disadvantage this behavior and a reward coefficient to the keeping lane cost (cuurrent) to advantage this behavior for more safety

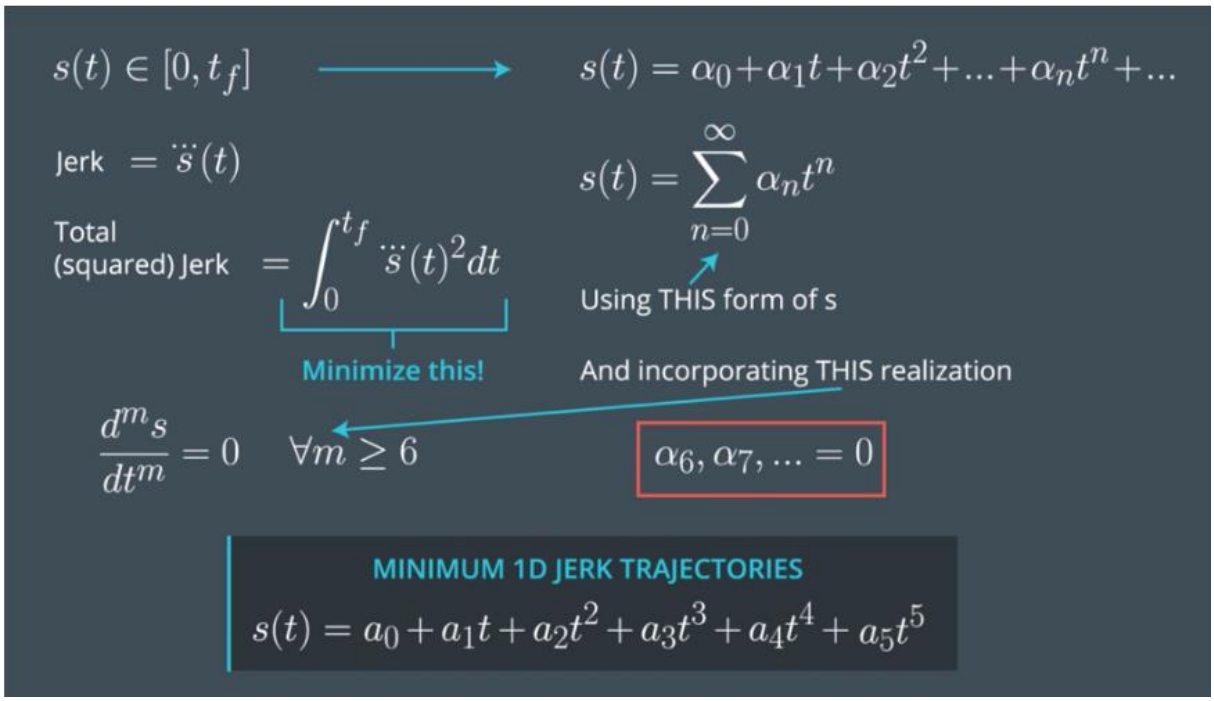
Trajectory Logical Component:
The Trajectory component is called by the main component make decision on lane change



Attributes :
1. **jmtpair** : it stores the s state related jerk minimum component and d state one

Methods :
1. **update** : This method computes the target velocity according to the legal class and safety class (see the code for more detail about the algorithm). It computes also the target s state as part of the data preparation process to finaly generates two JMT component s JMT and d JMT. Both of them will be stored in the jmtpair attribute for later invocation by the main component.

JMT Logical Component:
The JMT stands for Jerk minimum trajectory component. In my design it is called by the Trajectory component minimize the jerk on trajectory defined by the start state and the target state. This object represents a quintic polynomial function of a number which has six coefficients. Once instantiated, you can evaluate this polynomial by giving a value. Jerk is the instantaneous change in acceleration over time like acceleration is the instantaneous change in speed over time. Riding with jerks is really uncomfortable. Applying the confort class to acceleration is minimizing the jerk.



Attributes :
1. **jmtpair** : it stores the s state related jerk minimum component and d state one

Methods :
1. **JMT (Constructor)** : This method implements formula (seen in course) to get the polynomial path position function that minimizes the squared jerk to get from the start state to the target state given the duration.

PathTransform Logical Component:
This commponent has a representation of the global map of the highway. It takes two `JMT` components which represents the desired path. This is something the controller does not understand in oposite to discrete points along the map in cartesian coordinates.

Attributes :
N/A

Methods :
1. **compute** : This method transform both of longitidinal and lateral paths in frenet coordinates into discrete points when we give it the distance between points along the path and number of points.

2. **frenetToxy** : This method is called by the compute method to convert the JMT paths (longitudinal and lateral) in frenet coordinates to paths in cartesian coordinates. To do that, it uses the spline component to fil the cubic polynomial and calculate the discrete points.