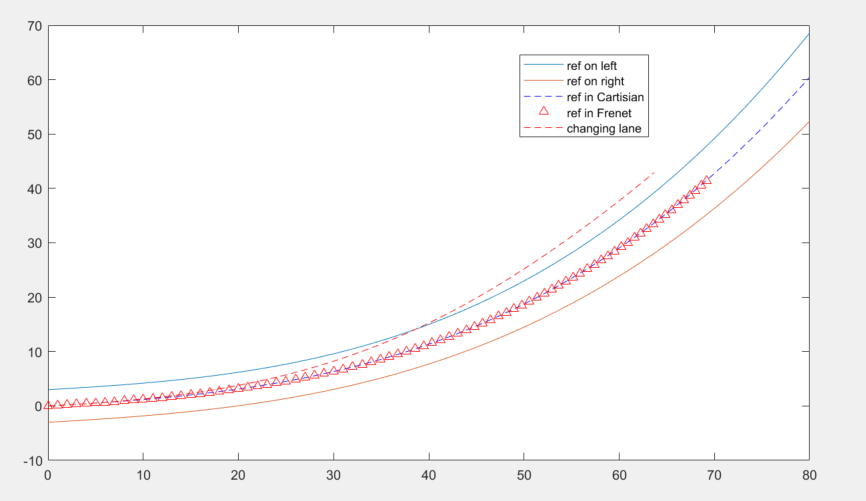
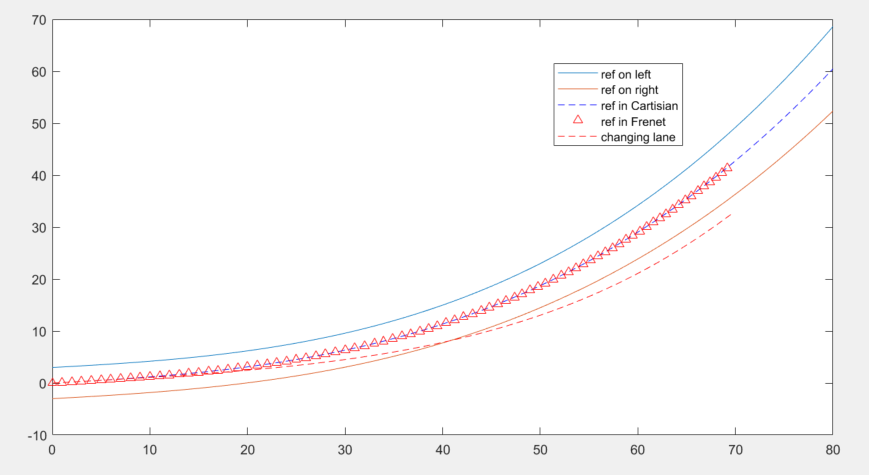
Test report

This report offers the test result in trajectory generation functionals.

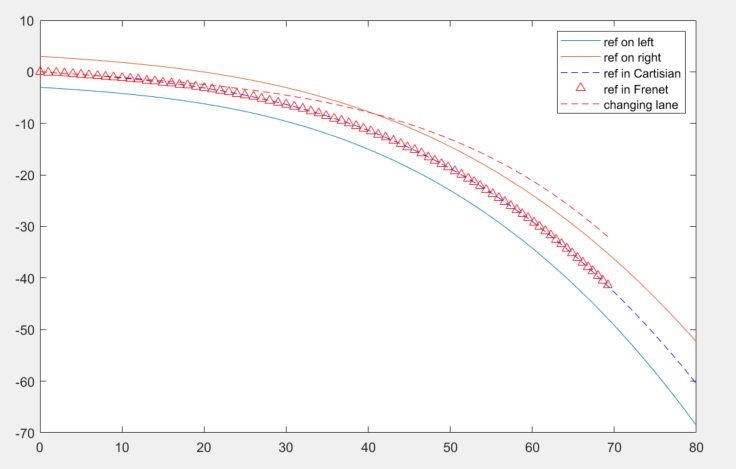
1. Lane changing functions test
2. Lane change trajectory generation test. We generate the four cases where we apply lane change functionals.
   1. Vehicle turning left, with lane changing left



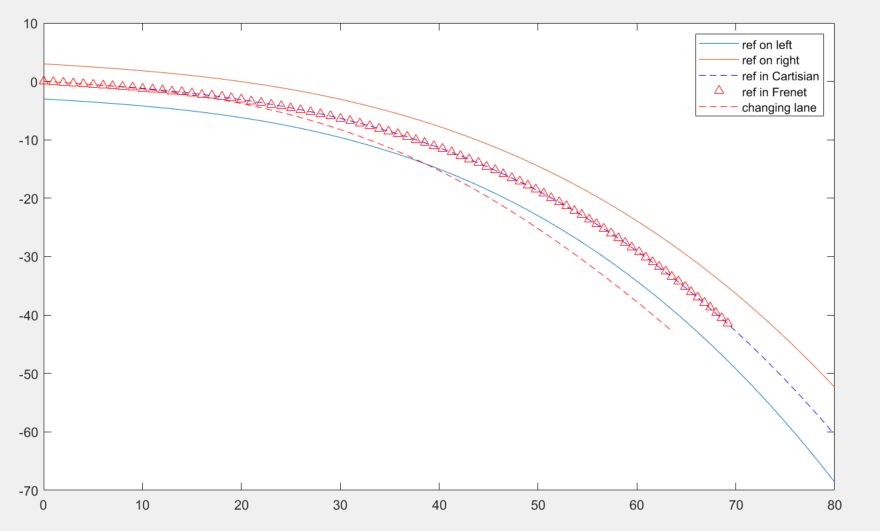
* 1. Vehicle turning left, with lane changing right



* 1. Vehicle turning right, with lane changing left



* 1. Vehicle turning right, with lane changing right



1. Comparison between two different center line generation

We have two difference methods in generating a sequence of center line waypoints for future use. One is, based on current polynomials of lanes on the left and right, discretely sample on Cartesian coordinate along x axis to get the y axis value, and calculate its mean value. And the other method is discretely sample under Frenet coordinate, which means it integrate along s direction. At each sample point, get the respective x value on each lane polynomial. We provide both methods realization and make comparison numerically.

* 1. Using Cartesian coordinate to generate a sequence of center line waypoints.



* 1. Using Frenet coordinate to generate a sequence of center line waypoints.



As you can see in the figure, the upper triangle line is using Frenet method, while the blue dashed line is using Cartesian method. They almost overlapped. As for the calculation time, we also make a comparison as follows:

The prerequisite is that both methods generate a sequence of trajectory of 80m, and the sampling resolution is about 0.2m, which means about 400 points will be generated. The calculation time is (with CPU performance i7-9750H @2.60GHz and RAM 32G):

Using Cartesian method:

*reference\_1 generation time is: 1.680400 ms*

Using Frenet method:

*reference\_2 generation time is: 4.659700 ms*

1. Kappa calculation

In our module, we provide two difference kappa calculation methods, one is deducted by pure math, while the other uses discrete interpolation.

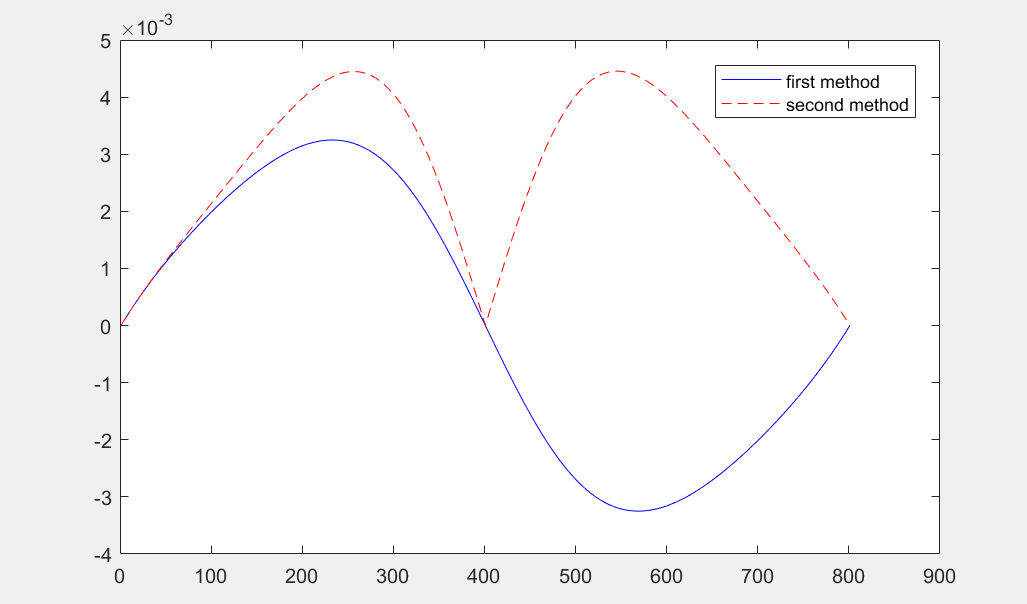
The first one use close form equation as follows:

The other method is as follows:

where

and

In the next few steps of path smoothing, we decided to trust the first method. And the two methods comparison is as follows:



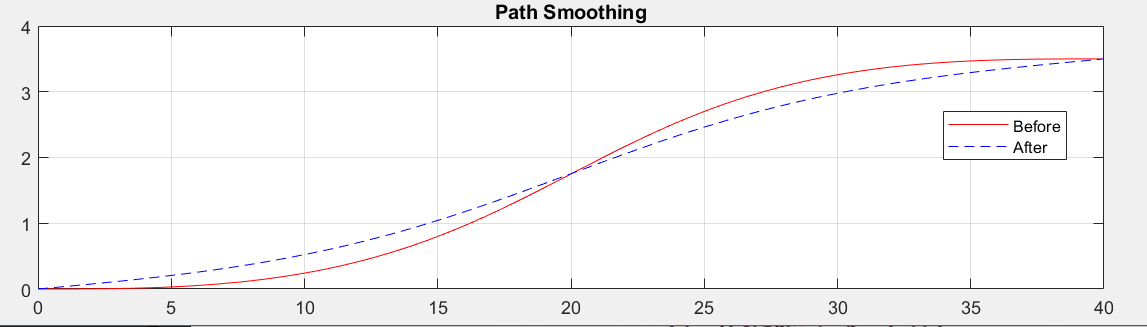
1. Path smoothing method:

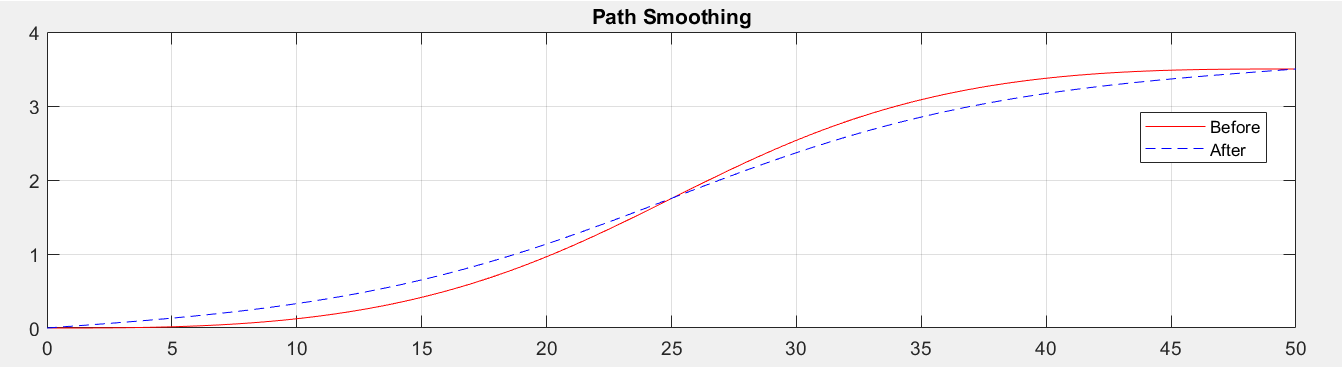
Since the original trajectory generated by Bezier curve cannot guarantee that the curve can satisfy the vehicle kinematic constraints, say, the max curvature constraints. Therefore, we designed a path smoother which aims to minimize 1) the distance between the origin reference line and the optimized trajectory. 2) the smoothness of the optimized trajectory.

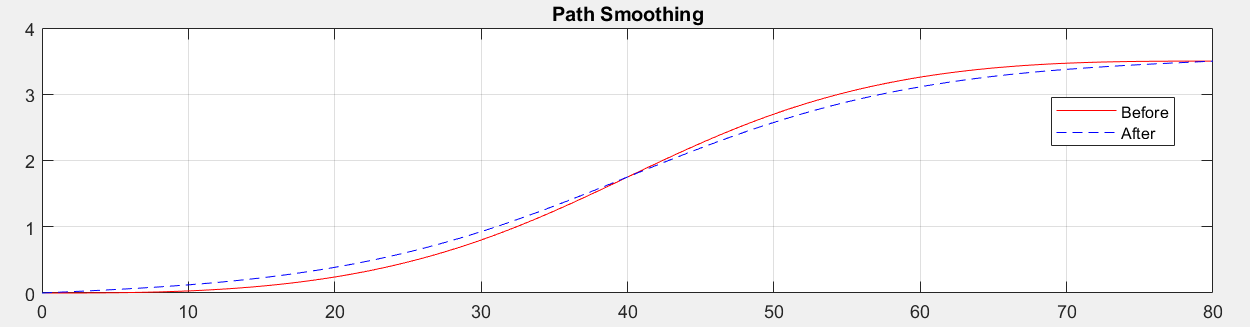
In this smoother, we penalized the norm of the offset between reference line and optimized trajectory, and the norm of . The objective function is as follows:

We use gradient descent method over to minimize objective function by iterative search.

We also add some stopping criteria to ensure the robustness of the method. The first term is constrained by add a maximum offset between the reference line and optimized trajectory, one the offset increases upper to the upper bound, the iteration will be forced to stop. In terms of curvature, we add a maximum curvature constraint as well. In each iteration, we will check the curvature of the optimized trajectory, once all the points on the trajectory satisfy the curvature limit, iteration will also be stopped. To fix the running time of the iterative search part, we lastly constrained the maximum iteration number as a constant. Several examples of the smoother performance is as follows, where we are smoothing the lane changing reference line, with lateral offset 3.5m, and longitudinal variation between 50m to 80m.







1. Sampling method

In this section, we discuss our method of candidate trajectory generation method which is based on lattice sampling (kind of), notice that we do not precompute the lattice heap matrix.

1. Bezier curve sampling method
2. Quintic polynomial sampling method