Derek Wu 08/19/2017
Term 2, Project 5

Student describes their model in detail. This includes the state, actuators and update equations.

First, I define the state in main.cpp.

State

$$[x_1, y_1, \psi_1, v_1, cte_1, e\psi_1]$$

Afterwards, the state coordinates is transformed into the car reference frame. This allows us to set some of the model equation to zero which simplifies the model. The transformed data points are passed into the polyfit code provided in class. The data points are fit to the third polynomial as suggested in the videos in class.

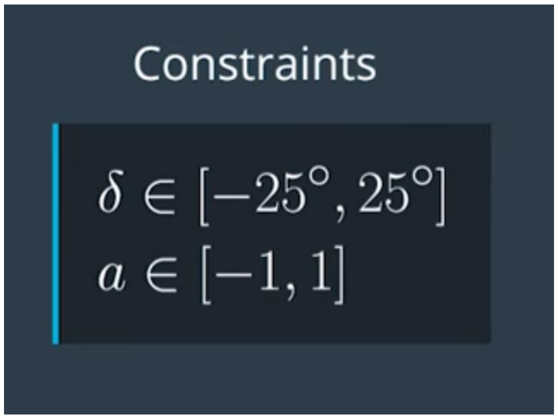
By fitting the lines to the polynomial, the cross track error and steering angle error can be calculated.

Then, according to the MPC model, the next state of the car is predicted using the model shown below:

Model

$$x_{t+1} = x_t + v_t * cos(\psi_t) * dt$$
$$y_{t+1} = y_t + v_t * sin(\psi_t) * dt$$
$$\psi_{t+1} = \psi_t + \frac{v_t}{L_f} * \delta_t * dt$$
$$v_{t+1} = v_t + a_t * dt$$
$$cte_{t+1} = f(x_t) - y_t + v_t * sin(e\psi_t) * dt$$
$$e\psi_{t+1} = \psi_t - \psi des_t + \frac{v_t}{L_f} * \delta_t * dt$$

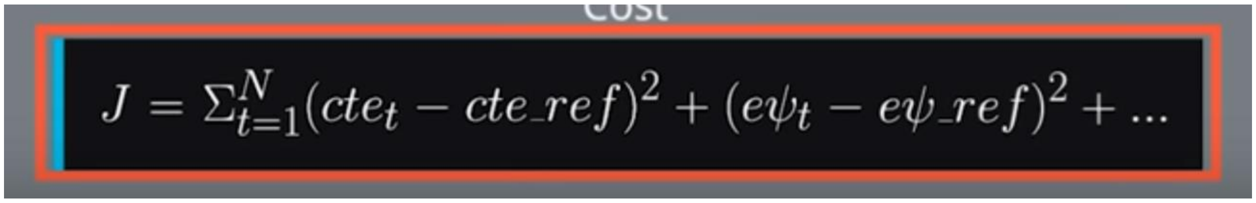
The estimated next state and the fitted polynomial is passed into the MPC.cpp function. The MPC function uses the guideline and recommended library to minimize the cost function and obtain the appropriate actuator responses. In order to do so, the code first set some constraints as suggested below:



Constraints

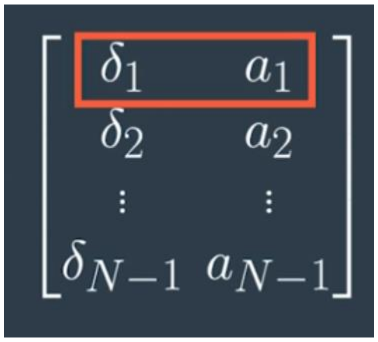
$$\delta \in [-25^\circ, 25^\circ]$$
$$a \in [-1, 1]$$

Next, the cost function is defined as shown below:



$$J = \Sigma_{t=1}^{N}(cte_t - cte\_ref)^2 + (e\psi_t - e\psi\_ref)^2 + \ldots$$

The predicted state, cost function, and constraints are fed into the suggested library (CppAD and IPOPT) to minimize to cost function. The libraries and solver will output a solution in a matrix similar to shown below:



$$\begin{bmatrix} \boxed{\delta_1 \quad a_1} \\ \delta_2 \quad a_2 \\ \vdots \quad \vdots \\ \delta_{N-1} \quad a_{N-1} \end{bmatrix}$$

This information is fed into the actuator and the car moves. This process is continuously repeated throughout all waypoints.

Derek Wu 08/19/2017
Term 2, Project 5

Student discusses the reasoning behind the chosen *N* (timestep length) and *dt* (elapsed duration between timesteps) values. Additionally the student details the previous values tried.

The final results in the software was N=10 and delta_t = 0.1. The final result for delta_t was chosen because of 1/N is the period between each update. 10 update points per second was ideal since the expected day is around 100 ms for the actuator. Higher values were tried (such as N=20, delta_t = 0.05) but it did not do much since the actuator cannot response correctly. If N was changed below 10, the car is not responding as quickly as it can which causes the response time to drop.

Student provides details on how they deal with latency.

The latency of the actuator can be handled in the kinematic model as shown before. The latency of the actuator only effects the response time of the car. The kinematic model will be effected by the discretization of the points. If the latency is too high, the points will be too far apart causing the car to move “choppy”. However, the kinematic will still have accurate predictions at the specific calculated way points and the “choppiness” of the car can be tuned with heavier weights during the initialization of the reference state in the MPC algorithm which changes the interpolation of the route between points.