Model-Predictive-Control Reflection

The algorithm

1. Set N and dt.
2. Fit the polynomial to the waypoints in line 115
    coeffs = polyfit(ptsx_transform, ptsy_transform, 3)

3. Calculate initial cross track error and orientation error values using polyeval(coeffs, 0) & -atan(coeffs[1]);

4. Define the components of the cost function (state, actuators, weights etc). This is done in MPC FG_eval class

5. Define the model constraints.

N & dt

N is 20 while dt is 0.1.
This selection enables 2 seconds prediction into the future without a complete change of the scene while 100ms enables continues update of the actuators while 2 seconds is not too far into the future to cause.

Polynomial Fitting

A polynomial is fitted to waypoints is done using the polyfit function in main.cpp we are using a 3rd degree polynomial – 4 coefficients.

MPC Model

MPC Model is implemented in lines 129-134 to predict the state of the car in 100ms according to the MPC model of the following equations

$$x_{t+1} = x_t + v_t * cos(\psi_t) * dt$$

$$y_{t+1} = y_t + v_t * sin(\psi_t) * dt$$

$$\psi_{t+1} = \psi_t + \frac{v_t}{L_f} * \delta * dt$$

$$v_{t+1} = v_t + a_t * dt$$

Parameter tuning

The initial set of parameter set provide in the walkthrough didn't work well. Although my initial assumption was to placing the highest value on CTE it seems that it caused the vehiecle to overcompensate and get out of the lines.

Also the model has a problem in the speed setting, the model passed the reference speed set, increasing the speed weight didn't help only lowering the speed made the speed constraint effective when the speed difference multiplied by the weight was high enough. Setting reference velocity to 30 limited the speed to 65.