Model

I have used a model same as the lectures .As far as cost function is concerned I have used the following:

```
// The part of the cost based on the reference state.
 for (int t = 0; t < N; t++) {
   fg[0] += 6.5 * CppAD::pow(vars[cte_start + t], 2);
   fg[0] += 0.25 * CppAD::pow(vars[epsi_start + t], 2);
   fg[0] += 0.25 * CppAD::pow(vars[v_start + t] - REF_V, 2);
 }

 // Minimize the use of actuators.
 for (int t = 0; t < N - 1; t++) {
   fg[0] += 55000 * CppAD::pow(vars[delta_start + t], 2);
   fg[0] += 17* CppAD::pow(vars[a_start + t], 2);
 }

 // Minimize the value gap between sequential actuations.
 for (int t = 0; t < N - 2; t++) {
   fg[0] += 0.01 * CppAD::pow(vars[delta_start + t + 1] - vars[delta_start + t], 2);
   fg[0] += 0.00001 * CppAD::pow(vars[a_start + t + 1] - vars[a_start + t], 2);
 }
```

The cost operation are based on trial and error. I would like to know a more robust way to fund the cost function.

Timestep Length and Elapsed Duration (N & dt)

I used values similar to one in lecture and did not experiment too much with these.

Polynomial Fitting and MPC Preprocessing

As for preprocessing we need to turn observation to vehicle coordinates and then fit 3 degree polynomial .

Model Predictive Control with Latency

I have not done any specific handling for delay in the model.