Model Predictive Control Project

Rubric Points

- **The Model**:

  In this project, kinematic model is used which is based on vehicles kinematics. The kinematic model takes into consideration the following:

  1. Vehicle's x-coordinate(x)
  2. Vehicle's y-coordinate(y)
  3. Vehicle's orientation (ψ)
  4. Vehicles velocity(v)
  5. Cross-track error(cte)
  6. Orientation error($epsi$)

  as state. These are described by the following equations:

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

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

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

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

$$cte_{t+1} = f(x_t) - y_t + (v_t * sin(e\psi_t) * dt)$$

$$e\psi_{t+1} = \psi_t - \psi des_t + (\frac{v_t}{L_f} * \delta_t * dt)$$

  Here *t* denotes the previous time-step and *t+1* denotes the current time-step. The model uses the above equations to calculate the state [$x, y, \psi, v, cte, e\psi$] and actuations (throttle and steering angle) for the current time-step from the previous time-step.

- **Time-step Length and Elapsed Duration (N & *dt*)**:

  The values chosen for N and *dt* are:

N	*dt*
20	0.05

  This combination of values was chosen by trial and error method. The combination was chosen because it produced considerably stable behavior of the vehicle.

- **Polynomial Fitting and MPC Preprocessing**:

  A three-degree polynomial curve is chosen because two-degree polynomial would not fit properly to the curves and zig-zag turns of the road, and a four or five-degree polynomial may lead to over-fitting as well as overhead computational cost.

  The waypoints are preprocessed by transforming them to vehicle's coordinate system (lines *105 – 110* in *main.cpp*). This is done because now fitting the polynomial to the waypoints becomes simple as the vehicle is at origin (0,0) and orientation angle ψ is zero.

- **Model Predictive Control with Latency**:

  The MPC handles a 100 milliseconds latency delay. This delay is introduced in the beginning, before calling MPC::Solve() in lines 123 – 129 of *main.cpp*. We estimate the new state [$x, y, \psi, v, cte, e\psi$] of the vehicle after 100milliseconds and then proceed with calculations in MPC::Solve() so that we predict the actuations after 100 milliseconds.

An alteration in the equation of epsi which, as given in the lesson as

$$epsi = (psi0 - psides0) - v*delta/Lf*dt$$

is changed to

$$epsi = (psides0 - psi0) - v*delta/Lf*dt$$

in accordance to this thread in discussion forum.