Rubric Points

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

The kinematic model takes the certain information from the simulator: *ptsx, ptsy* which are x and y value of waypoints on the track in global coordinates, *px, py* which describes position of vehicle in global (simulator) coordinate system, *psi* which is current orientation angle (heading) of the vehicle, *v* which is current velocity of the vehicle, *delta* which is current steering angle value of the vehicle and *a* which is current throttle value. The model combines the state vector and actuations vector from the previous timestep and calculate current state and the actuations vector using equations below:

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

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

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

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

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

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

2. 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 values chosen for *N* and *dt* are 10 and 0.1, respectively. I originally used values of 10 for *N* and 0.3 for *dt*. I thought 3 second would be a good prediction span. However, I found 0.3 to be way too slow to react. I also noticed that model slows down if *N* was higher than 10 so I eventually settled on 10 for *N*, which meant that with 0.1 *dt.*

3. A polynomial is fitted to waypoints. If the student preprocesses waypoints, the vehicle state, and/or actuators prior to the MPC procedure it is described.

The waypoints were preprocessed by transforming them to the vehicle's perspective. After transformation the vehicle's *x* and *y* coordinates are at the origin (0, 0) and the orientation angle is also zero which simplifies fitting a polynomial to the track waypoints.

4. The student implements Model Predictive Control that handles a 100 millisecond latency. Student provides details on how they deal with latency.

The main idea to dealing with latency was use the same update equations as those used in the actual MPC model to predict the state (predict where the vehicle would be after 100ms) and put new state in solver. Equations due to vehicle coordinates transformation are simplified. Also, in addition to the cost functions suggested in the lessons I implemented additional cost penalizing the combination of vehicle velocity and steering angle. This additional cost slightly improve vehicle cornering.