1. The model.

    The state consists of 6 variables:
    A. X position
    B. Y position
    C. Vehicle orientation Ψ
    D. Vehicle velocity v
    E. Cross track error CTE
    F. Orientation error eΨ

    Actuator inputs consists of the following.
    A. Steering angle δ
    B. Acceleration a (which accounts for breaking as well)

    Update equations.

    - X position update is given by the following
      $x_{t+1} = x_t + v_t cos(\psi_t).dt$

    - Y position update is given by the following
      $y_{t+1} = y_t + v_t sin(\psi_t).dt$

    - Velocity update
      $v_{t+1} = v_t + a * dt$

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

Where $L_f$ measures the distance between the front of the vehicle and its center of gravity.

2. I used N = 20 and dt = 0.05. This was optimal with respect to the trajectory followed and simulation performance.

N determines the number of timesteps in the horizon and dt determines the time elapsed between actuations. The product of N and dt gives the time horizon T. Keeping a very high value of T is of no use, since the environment will change enough that it wont make sence to predict into the future.

I first tried with N = 5 and dt = 0.05. The car quickly went and hit the curb. Mostly the horizon was too short.

Then I tried with N = 20 dt = 1. The car was moving too far from the center throughout.

Finally with N = 20 and dt = 0.05 the car was moving roughly at the center of the lane.

3. Preprocessing prior to fitting polynomial.

- Convert velocity to m/s as the kinematic equations require it in this unit
- Convert vehicle position from simulator coordinates to vehicle coordinates.

4. Latency handling :

To handle latency, the state of the vehicle was predicted using the kinematic equations and current state for the given latency. This predicted state was then sent to the solver to obtain the trajectory, which would be more accurate.