MPC Project

Mandar Joshi

1. **Code Layout & modifications**
    a. MPC.cpp/h and main.cpp
    b. radiusOfCurvature() defined in Main.cpp. Used to compute the radius of curvature of waypoints at a given location of the car.
    c. A DEBUG macro was added to MPC.h
2. **Result**
    a. The car was able to successfully navigate the track at a max speed of 93/94mph. The speed was throttled based on the curvature of the road.
    b. Please see the video here - https://youtu.be/UqxxRdR8kh8
3. **Process, Model**
    a. We receive the following information about the car from the simulator in a loop
        i. Waypoints in ptsx, ptsxy - map coordinate system
        ii. (x, y) location of the car in map coordinate system
        iii. psi, velocity v, steer_angle s, and acceleration a
    b. First, we take into account the latency of the system (100msec) by predicting the position of the car in 100msec and use that as the position of the car from where the actuator values need to be determined. This is done in the map coordinate system, so that we can feed the location into the next step that converts to vehicle coordinate system with the car at the origin (0,0)
    c. Second, we convert the waypoints into the vehicle coordinate system in ptsx_v and ptsy_v.
    d. We then fit a 3rd order polynomial to the transformed points and compte the new cte, epsi
    e. Before we call the MPC solver to compute the optimized path forward, we compute the radius of the curvature of the path dictated by the waypoints. This is done on the waypoints in the map coordinate system since that would be the most accurate measurement of the curvature. To do this, we first fit a polynomial to the waypoints in map coordinate system and save the computed radius of curvature.
    f. We then call the solver to compute an optimized path forward and the associated actuator values.
    g. Model:
        i. State:
            1. The state provided is in the vehicle coordinate system. So the position of the car is (0, 0) and orientation psi is 0 as well.
            2. The coeffs computed by fitting a polynomial to ptsx_v, ptsy_v in the

vehicle coordinate system is provided as input to the solver.

ii. N & dt:
    1. N is the number of dt time steps we would like for the solver to "see" into the future
    2. N*dt is the total time duration into the future the algorithm will predict
    3. A large N is computationally expense. However a very small N may not be sufficient for the algorithm to see enough into the future and come up with optimal values for the actuators.
    4. The required speed has an impact on the selection of N and dt. In this implementation the reference speed was set to 100 mph. The optimal value for N and dt was determined to be 13 and 0.04 seconds.
    5. Several other combinations were tried. Some notable ones:
        a. 10 and 0.1 – at high speeds this would often result in a solution on turns where the mpc trajectory would be quite a bit different than the waypoints. This was most likely due to the high weight on optimization of sequential delta and would result in non optimal solutions.
        b. 20 and 0.05 – This resulted in predictions that were further out and would work ok at lower speeds but would fail as the speed was increased.

iii. Solver setup: There are various inputs that need to be setup for the solver.
    1. Vars: This variable contains all of the state variables and actuator predictions based on the value of N. There are 6 state variable and 2 actuators. So the size of vars in this implantation is 13*6 + 12*2 = 102
    2. Vars_lower/upperbounds: For the state variables and actuators, this provides the upper/lower bounds to maintain towards a optimal/lowest cost solution. For delta this is set to be +/-25 degrees and for acceleration +/-1.
    3. Constraints_lower/upperbounds: This is not very critical from the perspective of the optimization that we are looking for since we do not have specific constraints as a combination of the variables above. However, the initial value of the variables must be set to the current state of the car.

iv. Cost Function: We use the following
    1. Minimize the cost based on reference state
        a. Minimize the cross track error
        b. Minimize error in psi
        c. Minimize the difference between the current speed and set speed limit, so as to try and maintain the speed at the reference value of ref_v
    2. Minimize the value gap between sequential actuations
        a. Minimize steering value change between two sequential actuations. This helps in dampening the oscillations of the car

and was found out to be one of the most optimization function. The result was a very high multiplier value of 80000.

b. Minimize the acceleration different between two consecutive actuations. This results in a smooth change in acceleration.

3. Minimize the use of actuators: This helps in dampening the effect of the actuator.
    a. In spite the above optimization, it was found that delta value should be dampened to further optimize the results.
    b. Acceleration: We use the radius of curvature (computed earlier) to dampen the acceleration actuator. Low radius of curvature indicates a sharp turn and so must result in the acceleration being reduced. High radius of curvature indicates straighter road, diminishing the need for the speed to be throttled. So, they are inversely proportional. A multiplication factor is used to make the resulting weight rs to be meaningful.

v. Equations:
    1. Our function computes the cost function based on the variables provided by the optimization library CppAD
    2. We also walk through the states provided by the library and see how it matches with our vehicle model i.e. for the variables provided we use the vehicle model equations to compute the state and compare it to the one sent to us by the library. The idea is to constrain the difference to 0.
    3. This cost function and the difference above is then used by the library towards creating a cost optimized functions based on the variable bounds and constraints. This function ends up being called multiple times until it determines that it has a cost optimal solution.