In our framework, CarSIM is a “high-order nonlinear longitudinal vehicles model”, regarded as a realistic world. The input variable in this virtual world is the desired acceleration, also called “acceleration command”. There are many output values from CarSIM. For example, relative distance with front car, longitudinal velocity, realistic acceleration and so on. As for Python, it act as a client receiving information from CarSIM. Python will execute Model Predictive Control, optimize the desired acceleration each sampling time, and send back acceleration command to CarSIM.

Let us look more detail into optimization algorithm. we know MPC is a kind of complex controller. It take a lot of calculation to optimize the input value in each sampling time. Simplifying all of our equality constraints and inequalities constraints to linear equations can help MPC find a anser more effenciently. Also, because our objective function is quadratic form, we can solve our problem by QP solver. We chosed OSQP, a powerful QP solver available in the Python package. OSQP can ensures an effective and efficient solution to our control problem. This enables us to execute the optimization process seamlessly at each sampling time.

In the initial phase of Experiment 1, we want to test the performance of the ACC system during cruise mode. The simulation started with the CarSIM model operating without ACC, navigating at an acceleration of 0.5m/s² for the roughly first 30 seconds. However, as illustrated in the figure, the CarSIM vehicle failed to meet the specified requirements. The realistic acceleration fluctuations significantly during this period.

Upon about 30-second, the ACC system was activated. It seamlessly transitioned into cruise mode, promptly issuing appropriate acceleration commands to the CarSIM model. As the graphic show, the acceleration of the CarSIM model exhibited a smoother transition with minimal error. Furthermore, the vehicle's speed remained consistently stable at 80 kph.

As for Experiment 2, we want to test cruise control mode and following mode in ACC. About 25 seconds ago, the vehicle was traveling with an acceleration of 0.5 m/s². After this, the ACC system is activated, and goes into cruise control mode with the cruising speed is set to 70 km / h. But when the front car was detected in the safety distance at about 42 second, ACC switch in to following mode. From the figure, we can see that vehicle is trying to maintain the distance and relative speed with front car.

MPC will do the optimization with constraints within time horizon p at each sampling time. In other words, each optimization have to consider plenty of variables and constraint equations due to different time steps. This make us troubled when checking the necessary conditions in those constraints. So we adjust the method trying to find the active constraint. By removing the constraints one by one, we can test the activity of those constraints. But, We found that each constraint removement will affect the simulation results. It seems that different constraint may be violated at different time step k.

After testing many times, we noticed that the performance of the Model Predictive Control (MPC) optimization algorithm was highly related to the set parameters. If the weight parameters for different desigen variables in objective functions were changed, the result may led to collision.

ACC takes care of the vehicle speed, automatically adjusting and ensuring a safe following distance from the vehicle ahead. This not only enhances safety but also results in reduced fuel consumption by minimizing frequent accelerations and decelerations.