Model Predictive Based Adaptive Cruise Control Project Report

by

Abhinav Gupta and Jingwei He

ME780 Introduction to Vehicle Dynamics, Control, Hybrid/Electric Vehicles, and Automated Driving

University of Waterloo

Spring 2025

Abstract

With the advancement of the advanced driver assistance systems (ADAS), adaptive cruise control (ACC) has become a quite popular research topic. In this work, the aim is to design a MPC controller that can be implemented in an ACC system. The work starts by reviewing the fundamental concepts and working principles of Adaptive Cruise Control (ACC) systems through performing a literature review on recent journals and articles. The literature review also covers the common control strategies and addresses possible future directions of development and research for this topic. Afterwards, an attempt was made to develop a functioning ACC system for the University of Waterloo Alternative Fuels Team's (UWAFT) Cadillac LYRIQ using the MPC control strategies. Throughout the development of the controller, a total of 4 different models were developed, simulated and compared: MPC with input and output constraints; MPC with input, output, and jerk constraints; and augmented model for control deviation based MPC with input, output, and jerk constraints implemented.

Table of Contents

Chapter 1: Introduction

Nowadays, with the advent of increased electrification and automation of vehicle controls, original equipment manufacturers (OEMs) have made strides to further automate road and highway driving for simple maneuvers such as maintaining a reference speed or tracking a safe headway distance when following a preceding vehicle. These specific actions, when fully integrated into a vehicle are known as the Adaptive Cruise Control (ACC) feature. Essentially, the test vehicle with ACC active utilizes sensors such as radar and/or camera to make leading vehicle detections and provide environmental feedback to develop appropriate torque and braking responses resulting in it slowing down or speeding up longitudinally to maintain a safe following distance or speed value set by the driver.

The ACC algorithm for this project will be formulated by solving an optimized controls problem within a model predictive controls (MPC) framework. The controller will be designed by basing it off of a longitudinal vehicle dynamics model during the development phase. This is to provide reliable autonomous path tracking in terms of minimizing errors to the reference without compromising driver comfort, while also designing it to be robust in the face of disturbances and modelling inaccuracies. Vehicle jerk is the main state that will be monitored for the driver comfort aspect. This controller will be designed with the goal of eventually integrating it into an actual vehicle, the University of Waterloo Alternative Fuels Team's (UWAFT) Cadillac LYRIQ, and thus parameters and actuator constraints relevant to the real-life vehicle will be used during system modelling and controller design.

As is the case with an MPC formulation, a cost function will be determined to meet the objectives set out for this control system given the actuator and comfort prescribed constraints. The cost objective for the MPC will be setup to minimize jerk states as well as maintain both following distances, based on a bounded following distance window, and driver set speed, depending on the mode/state the ACC feature is in.

Based on research and experimental results regarding performance as well as computation time, either an architecture incorporating MPC weight switching or explicit controller handoff based on feature state may be implemented. A lower-level PI controller may also be included for motor torque determination to compensate for inverter delay. The algorithm would be designed using the MATLAB and Simulink programs. It will be tested with environment feedback provided from the RoadRunner Scenario simulation platform.

In this work, a literature review is first performed to provide understanding of the working principles and major components of an ACC system, discuss the conventional control strategies that have been implemented in modern ACC systems and their advantages and shortcomings, as well as to address the current technological barriers and future perspectives of the ACC system. Followed by the literature review, the work then discusses the group's attempt to develop and simulate a MPC controller for ACC applications, where 4 different models would be addressed: MPC with no constraints, MPC with input constraints, MPC with output constraints, and adaptive MPC.

Chapter 2: Literature review

A. Introduction to Adaptive Cruise Control in Electric Vehicles

1. Background

In recent years, a majority of the vehicles have been equipped with advanced driver assistance systems (ADAS) in order to enhance driving comfort, driving safety, fuel efficiency, traffic capacity, as well as to reduce errors in driving. Out of all the ADAS options, the adaptive cruise control (ACC) systems have become a popular research topic.[1] Although the ACC systems were only available for high-end vehicle models when it was initially introduced in the Japanese market in 1995[2], it is commonly implemented on both high-end and mid-range vehicles and has been developed based on a variety of trends. For example, for the mentioned fuel efficiency enhancement purpose, energy-optimal adaptive cruise control (EACC) has been developed to reduce the fuel consumptions of vehicles through optimizing traction force and braking force.[3]

2. Major Components and Working Principles

Typically, an ACC system consists of four major components: sensors, actuators, human-machine interface, and controller.[4] The sensors include cameras, radar, and lidar. They are responsible for detecting driving environment conditions, such as information about a preceding vehicle, and sending it to the controller. The controller takes the input information to compute the proper throttle and brake commands. The actuators then take the commands and implement them to achieve an appropriate speed or following distance. The human-machine interface provides information on the vehicle's states to the driver and warns the driver in case human intervention is necessary.

Since ACC systems are an extension of the conventional cruise control (CCC) systems, they are often used in conjunction with CCC systems in vehicles. Essentially, the vehicle would switch between ACC mode and CCC mode depending on the driving environment conditions. When there are no vehicles present ahead of the host vehicle. The host vehicle remains in CCC mode and travels at a constant set speed determined by the driver. As the sensors detect a preceding vehicle, the ACC system will take over and control the speed of the host vehicle to maintain a safe following distance, which is determined by vehicle length, speeds of both vehicles, current distance between the two vehicles, and a constant-time headway for the driver to react. In the case that the preceding vehicle accelerates constantly such that the ACC outputs a following speed that exceeds the driver-set speed, the vehicle will switch back to CCC mode and travel at the set speed instead of accelerating further to follow the preceding vehicle. Table 1 below summarizes the logic of switching between the two systems, where v_{ref} is the driver-set speed, v_r is the velocity difference between the two vehicles, and d_{ref} is the safe following distance.

Table 1: Logical rules for switching between ACC and CCC

	($v<v_{ref}$)	**($v\geq v_{ref}$) & ($v_r<0$)**	**($v\geq v_{ref}$) & ($v_r>0$)**
$d\leq d_{ef}$	ACC	ACC	CCC
$d>d_{ref}$	CCC	CCC	CCC

3. ACCs in Electric Vehicles

The ACC system is widely installed on vehicles with internal combustion engines (ICE). However, when it is being installed on electric vehicles, the design might need to be modified so that it is related to the electric motor control block.[6] For example, for Battery Electric Vehicles (BEV), the state of charge needs to be considered when developing a control strategy for improving the performance. For Hybrid Electric Vehicles (HEV), where both the electric motor and the ICE collaborates to output the torque, a specially tailored control strategy needs to be designed to increase the efficiency of the Energy Management System (EMS).[4]

B. Conventional Control Strategies and Limitations

As for the controlling techniques that are used in the ACC systems, there are two main control structures. One is a one-level end-to-end controller. This kind of controller commands the required throttle or brake response to the actuators directly based on the detected state information from the sensors. The other control structure is a hierarchical controller. Such a controller consists of two control levels, a high-level controller or an outer loop controller, and a low-level controller or an inner loop controller. Essentially, the high-level controller is responsible for acquiring the proper acceleration or speed to maintain a safe following distance based on the detected state information from the sensors. The low-level controller focuses on the actuation aspect and controls the throttle and the brake to achieve the required acceleration or speed from the high-level controller. A variety of control strategies exist for designing an ACC system. Some common control strategies include Proportional-Integral-Derivative (PID) control, Sliding Mode Control (SMC), Linear Quadratic Regulator (LQR) method, and Model Predictive Control (MPC).

1. PID

The PID controller is one of the most popular control strategies used in the ACC systems due to its simplicity of implementation.[7] The PID controller relies on error feedback to regulate the speed and the following distance of the vehicle. It can fulfill up to three design specifications. The controller consists of three terms: the proportional (P) term, the integral (I) term, and the derivative (D) term. Each is calculated individually and then combined to calculate the throttle or brake command. The proportional term regulates the throttle and brake responses and reacts to the current speed error or distance error, which is defined by the difference between the current value and the desired value. The integral term eliminates the steady-state errors. The derivative term anticipates future errors through damping the oscillations. The main advantages of a PID controller are its simplicity, its real-time stability and the ease of tuning through several conventional methods, such as Zeigler Nichols, fuzzy logic and artificial neural networks(ANN).[7] However, it has a few limitations as well. The PID controller does not have predictive capability and it also struggles with nonlinearities such as aggressive braking. As well, the fixed gain limits the flexibility of this control strategy. A common strategy used is gain scheduling, where various operating points of the vehicle are calculated at different positions of the throttle or the brake, in order to obtain a set of linear models.[8] Overall, the PID controller offers simplicity, but the limitations may result in jerking and overshooting, affecting the driving comfort and ride quality.

2. SMC

The SMC method is a robust approach proposed in the 1950s. The SMC method establishes a hyperplane for control, also known as the sliding surface. The sliding surface is a hypersurface or manifold such that the system would behave according to the requirements when confined to this manifold. The SMC forces the states of the ACC system to converge onto and slide along the sliding surface and control them to a balanced point. The advantages of the SMC method include its quick response, insensitivity to parameter variations and disturbances, and its robustness against model uncertainties and perturbations. Hence, the SMC method has been implemented in the ACC systems of luxury or high-end vehicles.[4] Meanwhile, it also has two main drawbacks. First, it has high-frequency chattering from oscillatory throttle and brake jerking , which leads to riding discomfort. As well, the proper tuning of a SMC controller is quite challenging. Hence, although the SMC offers great robustness towards system uncertainties and external disturbances, it has high tuning complexity and lacks smoothness due to its limitations.

3. LQR

The LQR method is an optimal control strategy that utilizes state-space representations and computes the acceleration commands through solving a minimization problem that minimizes a quadratic cost function in an infinite horizon,[4]

$$J=0(xTQx+uTRu)dt \tag{1}$$

, where the Q and R are the weighting matrices. By properly defining the weighting matrices, it is possible to optimize for vehicle performance, ride quality and fuel efficiency.[9] The gain matrix can be evaluated through solving the algebraic Raccati equation that associates with the LQR problem. The LQR method is often used to deal with trade-offs between different performance indices, such as balancing the tracking errors and control effort.[5] The benefits of implementing the LQR method lies in its computational efficiency and stability, as well as its ability to address multiple performance indices. However, there are limitations to the LQR method as well. One drawback is its lack of ability to handle constraints, which limits its real-world implementation in the ACC systems. Besides, it has limited flexibility and effectiveness at handling nonlinearities due to the fixed gain matrix. Similarly, the gain scheduling strategy can be utilized on LQR controllers as well.

4. MPC

The MPC strategy is also known as Receding Horizon Control (RHC). The MPC strategy shares some similarities with the LQR method. Both strategies involve using the dynamics model of the vehicle and both solve for a minimization problem using the cost function.[5] However, while the LQR method solves the problem in an infinite horizon, the MPC model is a discrete system that solves the optimization problem at each fixed time step. The major working principle of the MPC is to predict the vehicle's future states through its current states in order to generate the proper response for achieving its objective. The error between the actual and predicted values can be minimized through using feedback. In other words, the throttle and brake commands can be adjusted and optimized based on the current vehicle states at each time step. Therefore, it has better flexibility and ability at dealing with nonlinearities. Similar to LQR, the MPC strategy is effective at balancing multiple performance indices through minimizing the cost function. Another advantage of MPC is its ability to handle constraints regarding velocity, safety, driving comfort, traction force, and braking force.[10] This enhances the smoothness as well as its real-world deployment. The shortcomings of the MPC strategy lies in its high demand in computation, which might eventually lead to hardware limitations.

C. Challenges of and Future Outlook

Although the MPC strategy has become the dominant control approach in designing ACC systems, there are still limitations and challenges regarding the deployment of MPC controllers. One limitation is in the hardware. As mentioned, the high computational complexity and demand of solving an optimized problem at each time step enforces strict constraints on the hardware of the system. In addition, the model can become inaccurate in dynamic environments, such as various friction coefficients. This hinders and degrades the performance of the ACC system. A possible future research focus can be on learning-augmented MPC. With future advancement in technology and embedded optimization, the implementation of MPC in the ACC system will accelerate, leading to features such as cooperative ACC (CACC), which can further improve traffic safety and energy efficiency by enhancing the predictions through vehicle-to-vehicle communications.

Chapter 3: System Modelling Methodology and Equations Formulation

A. Longitudinal Dynamics and Motion Modelling

In order develop a robust enough Model Predictive Controller for this Adaptive Cruise system which would be able to handle real world changes in states and minimize the errors as desired a dynamic longitudinal model of a vehicle is considered. The vehicle model is based on the parameters provided for the 2023 Cadillac LYRIQ Sports Utility Vehicle. The longitudinal dynamics equation is based on the formulation used in literature as well, notably the one presented in Pan's implementation of their optimal controls solutions [3] for this feature. First net force analysis is done on the vehicle body with a mass of *m* for the LYRIQ in order to determine the effect of an input force on the vehicle's acceleration a. Equation (1) below captures this:

$$ma = F_{trac.} - F_{drag} - F_{rolling} - F_{gravity} \quad (1)$$

where $F_{trac.}$is the net traction force propelling the vehicle forward, F_{drag} is the aerodynamic force opposing vehicle travel, $F_{rolling}$ is the rolling resistance the vehicle experiences, and $F_{gravity}$ is the force of gravity acting on the vehicle as a function of road grade. They each expand to the following definitions:

$$F_{trac.} = \frac{T_{axle}}{r_{wheel}} + ma_{brake}$$

where T_{axle} is the torque applied at the axle [Nm] by the electric motor to propel the vehicle forward, r_{wheel} is the tire radius, and a_{brake} is the deceleration value [m/s^2] that may be commanded from the stock vehicle braking Electronic Control Unit (ECU) to slow down its forward velocity. For resistance caused by drag:

$$F_{drag} = \frac{C_w \rho A_{wind} v^2}{2}$$

where v is the vehicle's forward velocity, C_w is the wind resistance coefficient that is specific to the vehicle's body, ρ is the density of air, A_{wind} is the windward area of the vehicle. And finally expanding the rolling and gravity resistance terms:

$$F_{rolling} = C_f mgcos(\theta)$$

$$F_{gravity} = mgsin(\theta)$$

where C_f is the rolling resistance coefficient, g is the acceleration due to gravity, and θ is the road grade angle. θ is assumed to be 0 degrees so the effects of $F_{gravity}$ cancel out to 0 N. The rolling resistance in this case would be a constant, however, to avoid having to include a constant disturbance when developing this controller, $F_{rolling}$ is also assumed to be negligible. Thus, the only external disturbance in the system left to be modelled is the drag term, F_{drag}. Since the velocity of the vehicle is squared, that makes this term non-linear. In order to present the effects of vehicle velocity on the experienced aerodynamic drag in a linear fashion, a nominal velocity value v_{nom} is chosen to multiply the second velocity term with. Thus, the drag term effectively becomes:

$$F_{drag} = \frac{C_w \rho A_{wind} v_{nom}}{2} v$$

Thus, combining all of these results back into equation (1) gives equation (1.5):

$$ma = \frac{T_{axle}}{r_{wheel}} + ma_{brake} - \frac{C_w \rho A_{wind} v_{nom}}{2} v \qquad (1.5)$$

However, in order to distinguish the *actual acceleration* experienced by the host vehicle (LYRIQ), a_h, from the *desired acceleration*, a_{des}, once the motor and brake pads are applied, the acceleration term in (1.5) will be renamed to a_{des} to give equation (2):

$$ma_{des} = \frac{T_{axle}}{r_{wheel}} + ma_{brake} - \frac{C_w \rho A_{wind} v_{nom}}{2} v_h \qquad (2)$$

where v is also renamed to v_h to reflect that it is the host vehicle's velocity. The torque applied by the electric motor and the brake force applied by the braking control unit are both assumed to have an actuator time delay modeled with τ_{delay}. Thus, the change in the host vehicle's velocity and acceleration are both modeled as shown in equation set (3):

$$\dot{v_h} = a_h$$

$$\dot{a_h} = \frac{a_{des} - a_h}{\tau_{delay}} \qquad (3)$$

B. Longitudinal Kinematics Modelling

Since the above equations deal with how the inputs to the system and dynamic disturbances affect the longitudinal motion of the vehicle, next to consider in modelling an ACC system is the “adaptive” component, which mainly refers to the lead vehicle speed following and safe distance tracking component of the feature. This aspect to the feature was included in the state-space modelling of the controller by relating the information available from both the long distance radar sensor on board the host vehicle as well as the information available about the speed and acceleration of the host vehicle. The radar sensor is able to provide the distance that is between the vehicle ahead and the host vehicle, denoted as $d_{following}$, and the relative velocity between the two vehicles as well, represented as v_{rel}. The error states are thus modelled as shown in equations (4):

$$\begin{aligned} \mathrm{d_{err}} &= d_{desired} - d_{following} \\ v_{rel} &= v_{lead} - v_h \qquad (4) \end{aligned}$$

where $d_{desired}$ is the desired following distance to be maintained from the vehicle in front and v_{lead} is the velocity of the vehicle in front. The desired distance calculation is done using the equation (5) taken from the paper by Pan [3]:

$$d_{\text{desired}} = 100 - 0.2(20 - 2.24v_h)^2 \qquad (5)$$

The rates of the change of the error states, d_{err} and v_{rel}, are modelled as shown in equations (6):

$$\begin{aligned} \dot{\mathrm{d}}_{\mathrm{err}} &= v_h - v_{lead} = -v_{rel} \\ \dot{\mathrm{v}}_{\mathrm{rel}} &= \dot{v}_{lead} - \dot{v}_h = 0 - a_h \qquad (6) \end{aligned}$$

As can be seen in equations (6) the acceleration of the lead vehicle is assumed to be negligible, and is thus modelled as a disturbance. Now there are enough equations to determine an appropriate state-space model for this system.

C. State-Space Model Formulation

The state vector, *x*, and input vector, *u*, chosen for this system are the following:

$$x(t) = \begin{bmatrix} v_h \\ a_h \\ d_{err} \\ v_{rel} \end{bmatrix}(t);$$

$$u(t) = \begin{bmatrix} T_{axle} \\ a_{brake} \end{bmatrix}(t)$$

Since a state-space system is generally of the form:

$$\dot{x(t)} = A_c x(t) + B_c u(t)$$
$$\mathrm{y(t)} = C_c x(t) + D_c u(t)$$

where $\dot{x(t)}$ captures the evolution of the chosen states over time with respect to each other, and *y(t)* is the output state vector. The continuous time matrices *A*_c, *B*_c, *C*_c, and *D*_c relating this set of chosen state and input vectors thus end up being:

$$A_c = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{C_w \rho A_{wind} v_{nom}}{2m\tau_{delay}} & -1/\tau_{delay} & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 \end{bmatrix}; B_c = \begin{bmatrix} 0 & 0 \\ \frac{1}{m r_w \tau_{delay}} & \frac{1}{\tau_{delay}} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

and

$$C_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; D_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

C_c being the identity matrix means that, through a simulated radar sensor on the vehicle and state feedback in Simulink from the vehicle dynamics block, the output vector is equal to the chosen state vector. In order to be used for an MPC formulation, however, the system needs to be discretized into the following representation:

$$x(k+1) = A_d x(k) + B_d u(k)$$
$$\mathrm{y(k)} = C_d x(k)$$

This discretization step is performed using the Euler method which determines an approximate discrete model of the system given a suitably small enough chosen timestep T_s [11]. Using the Euler method, the discrete state-space system matrices can be obtained as follows:

$$A_d = I + T_s A_C$$
$$B_d = T_s B_c$$
$$C_d = C_c$$

This results in the following discrete state-space system (7):

$$\begin{bmatrix} v_h \\ a_h \\ d_{err} \\ v_{rel} \end{bmatrix}(k+1) = \begin{bmatrix} 1 & T_s & 0 & 0 \\ -\frac{C_w \rho A_{wind} v_{nom}}{2m\tau_{delay}} T_s & 1 - T_s/\tau_{delay} & 0 & 0 \\ 0 & 0 & 1 & -T_s \\ 0 & -T_s & 0 & 1 \end{bmatrix} \begin{bmatrix} v_h \\ a_h \\ d_{err} \\ v_{rel} \end{bmatrix}(k)$$
$$+ \begin{bmatrix} 0 & 0 \\ \frac{T_s}{m r_w \tau_{delay}} & \frac{T_s}{\tau_{delay}} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T_{axle} \\ a_{brake} \end{bmatrix}(k);$$
$$y(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_h \\ a_h \\ d_{err} \\ v_{rel} \end{bmatrix}(k) \qquad (7)$$

The state-space system (7) is used as the basis for the rest of the model predictive matrix construction as well as state-space augmentation. While a model predictive controller is designed for the system determined by (7) above, due to the lack of emphasis over the change in applied control signal in the system above, a state-space model that describes the same dynamics but instead shows the evolution of its states in terms of the control signal deviation Δu is also derived to design a controller for the purposes of minimizing the change in host vehicle acceleration as well. This system will be called the augmented state-space system and is determined by first making the following explicit:

$$u(k) = u(k-1) + \Delta \mathrm{u(k)}$$

and then using this result in (7) to give (8):

$$x(k+1) = A_d x(k) + B_d[u(k-1) + \Delta u(k)] = A_d x(k) + B_d u(k-1) + B_d \Delta u(k) \qquad (8)$$

The expansion described by (8) is used to determine the form for the new augmented state vector $\xi(k) = \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix}$ resulting in the following discrete augmented state-space system described by (9):

$$\xi(k+1) = \begin{bmatrix} x(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} A_d & B_d \\ 0 & I \end{bmatrix} \xi(k) + \begin{bmatrix} B_d \\ I \end{bmatrix} \Delta \mathrm{u(k)}$$
$$y(k) = [C_d \quad 0]\, \xi(k) \qquad (9)$$

The resulting derived matrices in (9) are dubbed as the new *A*$_{aug}$*, B*$_{aug}$*,* and *C*$_{aug}$ system matrices for the discrete augmented state-space model.

D. Model Predictive Controller Batch Matrix Formulation

Within the model predictive controls framework, an optimization problem is solved. This controls problem is to be optimized at every timestep for a set of system output states predicted over a finite horizon in the future. The number of prediction steps, *N,* determines the size of the predicted output vector $Y(k_i) = \begin{bmatrix} y(k_i+1|k_i) \\ \vdots \\ y(k_i+N|k_i) \end{bmatrix}$ which is found as a result of the following matrix computation $Y(k_i) = Fx(k_i) + \phi U(k_i)$ where $U(k_i) = \begin{bmatrix} u(k_i) \\ \vdots \\ u(k_i+N-1) \end{bmatrix}$ is the full optimized control vector over the prediction window determined at a given time step. *F* and ϕ are likewise the so-called batch matrices that are pre-multiplied with the state vector at the current time step, *x(k$_i$),* and predicted set of optimized control vectors, *U(k$_i$),* respectively to determine the full prediction output vector. The batch matrices are to be determined as shown in the block matrix formulation (10):

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix} ; \Phi = \begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & & & & \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \dots & CA^{N_p-N_c}B \end{bmatrix} \tag{10}$$

where N_p and N_c are the prediction and control horizons respectively, both made equal to N in the MPC controllers developed for this report. The batch matrices in (10) are computed separately for the regular and augmented discrete state-space system models identified in this chapter.

To make this fit for the cost functional, the Q cost functional matrix for the output state vector is expanded to a block matrix the size of the prediction horizon N with all diagonal entries being Q. R, the cost functional matrix for the control vector is given a similar treatment, resulting $\overline{\mathrm{Q}}$ and $\overline{\mathrm{R}}$ respectively. To optimize the control vector the following H and f matrices (11) are computed:

$$\begin{aligned} H &= 2(\phi^T \bar{Q} \phi + \bar{R}) \\ f &= 2\phi^T \bar{Q} F x(k_i) \end{aligned} \tag{11}$$

The next chapter will detail the simulation implementation as well as specifics for the Model Predictive Controllers designed. Methodology for tuning the controllers and metrics for feature performance and actuator and state constraints will also be delved upon. Finally, an analytical and visual comparison of the regular and augmented discretized state-space based MPCs' performance results will be presented and discussed.

Chapter 4: Implementation Details and Simulation Results Discussion

A. Simulation Setup and Implementation Details

The code for the controllers is generated and implemented with the use of the MATLAB/Simulink software packages. A lead vehicle is simulated in the RoadRunner Scenario vehicle simulator, which is connected to the model predictive controls code housed in MATLAB function blocks in Simulink. Figure 1 below shows the Simulation in the Loop (SIL) setup with code in Simulink and the scenario with a lead vehicle in RoadRunner.

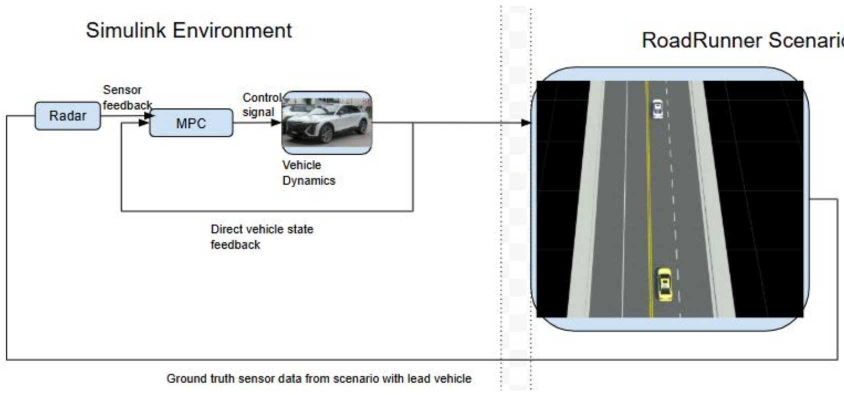

Figure 1: Block diagram of SIL setup

As shown in the block diagram above, the MPC controller sends an optimized control signal which is fed into a three degree of freedom dual track vehicle body dynamics block in Simulink. The vehicle dynamics block then provides direct state feedback based on its response to the control input to both the MPC controller and the host/test vehicle (the yellow car following behind) in RoadRunner scenario. The simulation in RoadRunner then relays real-time ground truth sensor data baed on the programmed trajectory for the lead vehicle (white vehicle in front being followed). The lead vehicle is programmed to have the following speed profile for the testing scenario: *5m/s for 2 seconds => 20 m/s at an acceleration of 2.5 m/s*^{2} *=> 27 m/s at 1.5 m/s*^{2} *=> 35 m/s at 3 m/s*^{2} *and then maintaining that speed for a total of 25 seconds => 0 m/s at a deceleration of -2 m/s*^{2}. This lead vehicle profile was kept the same for all testing done when developing these controllers as it covers a start from stop scenario for the host vehicle where the vehicle ahead of it also accelerates to varying speed increases with steady state acceleration and velocity characteristics, and then the lead eventually comes to a complete stop resulting allowing us to test the automated emergency braking capabilities of the tuned controller.

The vehicle dynamics block used is fed with the characteristics of the 2023 Cadillac LYRIQ summarized in Table 2 below. The timestep and horizon characteristics set for the Model Predictive Control implementation are also summarized in Table 2.

Table 2: Vehicle and Model Predictive Controller Characteristic Values

Characteristic	**Definition**	**Value**
N	Prediction and Control Horizon	20
T_s	Model and Simulation Sample Time	0.05 seconds
m	Mass of LYRIQ	2630.84 kg
τ_{delay}	Actuator Time Delay	0.2 seconds
r_w	Wheel Radius	0.378 m
C_w	Wind Resistance Coefficient of LYRIQ	0.30356
A_w	Windward Area of LYRIQ	2.73 m^2
ρ	Air Density	1.206 kg/m^3
v_{nom}	Nominal Vehicle Velocity	30 m/s
Driver Set Speed	Cruise Control Speed Limit	30 m/s

B. Constraints, Performance Metrics, and Cost Tuning

The actuators present in the vehicle have certain limitations, namely the maximum torque that can be commanded by the motors and the maximum deceleration that the braking controller can provide. Moreover, there are also output limits on the vehicle speed such that it should not go below 0 m/s (i.e. go in reverse) or exceed the set limit for cruise control. The acceleration also has a prescribed limit, along with the jerk states of the vehicle. Since jerk is what is "felt" the most by the driver in a vehicle, it is desirable to minimize this derived state as much as possible through a combination of applying constraints on your system and tuning the Q and R weight matrices. These limits are all documented in Table 3.

Table 3: Limits on Inputs, Outputs, and Jerk

Characteristic	**Lower Limit**	**Upper Limit**
Axle Torque (T_{axle})	0 Nm	4000 Nm
Braking Deceleration (a_{brake})	-3.5 m/s^2	0 m/s^2
Vehicle speed (v_h)	0 m/s	Driver Set Speed

Vehicle Acceleration (a_h)	-3.5 m/s^2	3.5 m/s^2
Vehicle jerk ($\frac{da_h}{dt}$)	-5 m/s^3	5 m/s^3

C. Results From MPC Implementation Based on Regular State-Space Model

1. Initial Testing and Tuning with Input and Output Constraints Implemented

The ACC model described by state-space system (7) yields the following A_d and B_d matrices once all of the parameters from Table 2 are substituted:

$$A_d = \begin{bmatrix} 1 & 0.05 & 0 & 0 \\ -0.0014 & 0.75 & 0 & 0 \\ 0 & 0 & 1 & -0.05 \\ 0 & -0.05 & 0 & 1 \end{bmatrix}; B_d = \begin{bmatrix} 0 & 0 \\ 0.0003 & 0.25 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}; C_d = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{12}$$

The system described by (12) is then used to determine the batch matrices from (10) and then compute the H and f matrices from (11) with current time step state vector, $x(k_i)$, provided through feedback in Simulink at each sample step. The input constraints are fed in as '*quadprog(H, f, A, b, [], [], lb, ub,* $U(k_i - 1)$*)*' where $ub = \overline{\mathrm{U_{max}}}$ and $lb = \overline{\mathrm{U_{min}}}$ such that $\overline{\mathrm{U_{max}}}$ and $\overline{\mathrm{U_{min}}}$ are the lower and upper limits for the actuators from Table 3 expanded to the size of the predicted control vector $U(k_i)$. The A and b parameters in the *'quadprog()'* function prototype above are used to express constraints in the form $AU(k_i) \leq b$. These are used to feed in the output constraints from Table 3 to the system as follows in form (13):

$$\begin{bmatrix} \phi \\ -\phi \end{bmatrix} U(k_i) \leq \begin{bmatrix} \overline{Y_{max}} - Fx(k_i) \\ -\overline{Y_{min}} + Fx(k_i) \end{bmatrix} \tag{13}$$

where $\overline{Y_{max}}$ and $\overline{Y_{min}}$ are the upper and lower limits expanded for the full predicted output vector $Y(k_i)$. Thus, the MPC implementation is first tested with the following weights: $q_{vh} = 0$, $q_{ah} = 5$, $q_{distError} = 100$, $q_{vrel} = 50$, $r_{Taxle} = 1$, $r_{abrake} = 1$. Since the host velocity state does not need to be minimized it is kept at 0, however, since two of the chosen states are errors that we want to minimize, namely the distance error and relative velocity, those are both set to be comparatively large. The inputs are kept at 1 start off to see their effect. The graphs in Figure 2 below show the system's response:

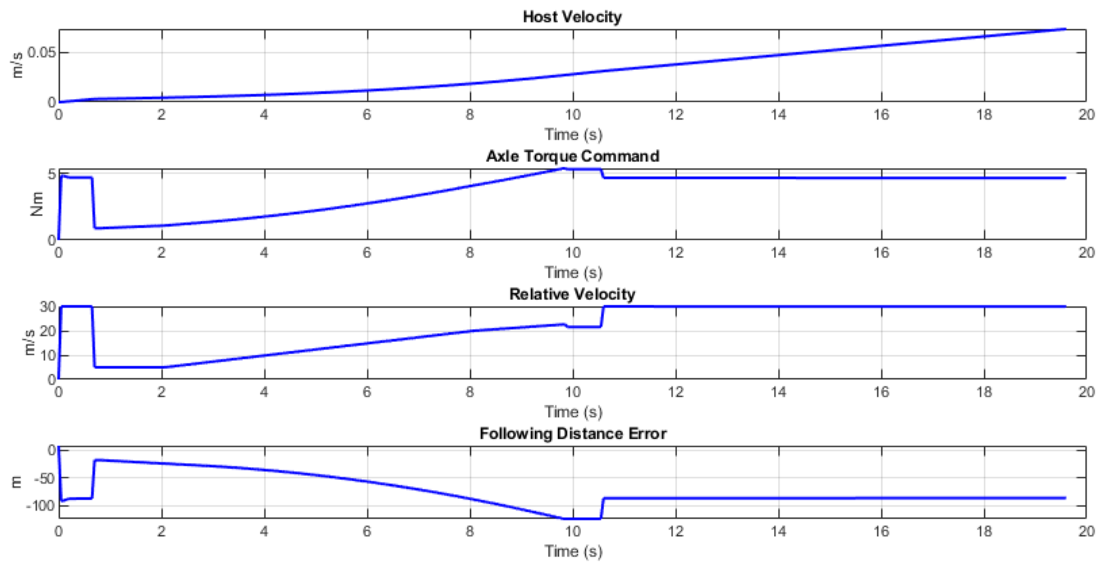

Figure 2: Host vehicle velocity, torque input, and erorr states with unaugmented MPC at weights $q_{vh} = 0$, $q_{ah} = 5$, $q_{distError} = 100$, $q_{vrel} = 50$, $r_{Taxle} = 1$, $r_{abrake} = 1$

As can be seen in Figure 2 the host vehicle barely moves, 0.05 m/s speed, and upon examination of the torque request plot it can be seen why. Since the torque required to move a vehicle is in the magnitude of thousands of Newton meters, the quadratic cost solver is attempting to minimize that value as much as possible. While it does make sense to minimize torque requests to avoid unnecessary accelerations, however, it needs to be weighted much less than the other states due to the nature of the command's interaction with the vehicle. Thus, despite the last two relative velocity and distance error graphs having fairly large values, around 30 m/s and -100 m respectively, which are unacceptable for real world performance, the cost objective believes a more feasible minimizing solution is still the one where axle torque is being commanded at near zero values. Thus, the cost weights are retuned to reflect this understanding. The new set of weights is set as follows: $q_{vh} = 0$, $q_{ah} = 20$, $q_{distError} = 100$, $q_{vrel} = 50$, $r_{Taxle} = 0.0005$, $r_{abrake} = 0.1$. The resulting graphs at this set of weights for the host vehicle's velocity and acceleration/jerk states as well as errors and inputs commanded are shown in Figure 3 and Figure 4.

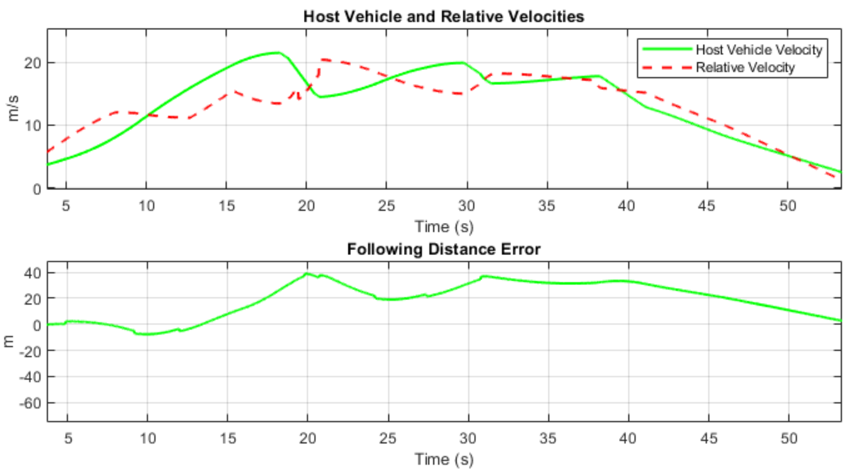

Figure 3: Host vehicle velocity and error states with unaugmented MPC at weights q_{vh} = 0, q_{ah} = 20, $q_{distError}$ = 100, q_{vrel} = 50, r_{Taxle} = 0.0005, r_{abrake} = 0.1

The relative velocity and following distance error plots from Figure 3 show that the controller is able to perform within the window of acceptability since the ACC feature is supposed to allow room for some degree of following distance and relative velocity errors as a real driver would in order to avoid large fluctuations in vehicle velocity. This is reflected in the acceleration and input signal plots in Figure 4.

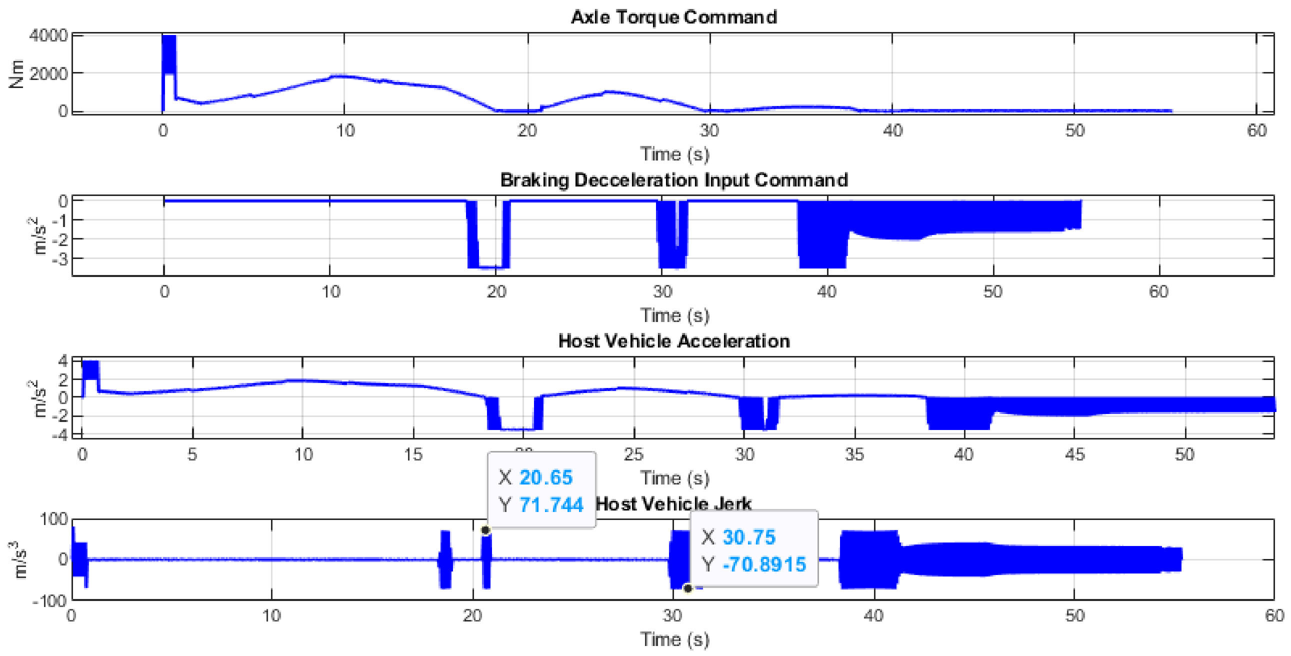

Figure 4: Host vehicle acceleration and jerk states and input command signals with unaugmented MPC at weights q_{vh} = 0, q_{ah} = 20, $q_{distError}$ = 100, q_{vrel} = 50, r_{Taxle} = 0.0005, r_{abrake} = 0.1

While it is good to note that the input output constraints are not being violated, the torque never exceeds 4000 Nm or go below 0 Nm, similarly, the braking deceleration is always kept at or above -3.5 m/s^2. Moreover, the third graph in Figure 4 shows that the acceleration of the vehicle itself also stays relatively within the prescribed constraint of ±3.5 m/s^2. While this means that the rate of change of the vehicle's velocity itself does not get very large, it says nothing of how often or drastically that value varies. In the second and third graphs of Figure 4 there are very large and frequent oscillations in the acceleration graphs, especially due to rapid and large jumps in the deceleration command. This makes it such that the fourth graph for jerk itself depicts large and frequent oscillations, with peaks reaching ±71 m/s^3. Thus, tuning the acceleration weight q_{ah} to be larger, as it was 4 times larger this time than in the previous iteration, does not help reduce the jerk of the vehicle.

2. Testing with Jerk Constraints Implemented

In an attempt to further limit this derived state, the jerk constraints are directly applied as an additional set of output constraints to (13), following the $AU(k_i) \le b$ form. Say that $jerk \le |J_{max}|$ is to be implemented. Simplifying the expression for jerk from equation set (3) to approximately be equal to $\dot{a_h} \cong \frac{\frac{T_{axle}}{r_w m} + a_{brake} - a_h}{\tau_{delay}}$. Thus, applying this jerk constraint to a control signal at a given time step looks like: $\begin{bmatrix} -\frac{1}{r_w m \tau_{delay}} & -\frac{1}{\tau_{delay}} \\ \frac{1}{r_w m \tau_{delay}} & \frac{1}{\tau_{delay}} \end{bmatrix} u(k_i) \le \begin{bmatrix} J_{max} \\ J_{max} \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} x(k_i)$. Let's denote $\Omega = \begin{bmatrix} -\frac{1}{r_w m \tau_{delay}} & -\frac{1}{\tau_{delay}} \\ \frac{1}{r_w m \tau_{delay}} & \frac{1}{\tau_{delay}} \end{bmatrix}$ and $M = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$, which would need to be expanded to be applied to the entire predicted control signal $U(k_i)$. Thus, we would need the entire predicted state vector, let's call it $X(k_i) = \begin{bmatrix} x(k_i) \\ \vdots \\ x(k_i + N - 1) \end{bmatrix}$, over this same window as well. Since, for this chosen state-space system the output vector is equal to the state vector (i.e. C_d is the identity matrix so $y(k_i) = x(k_i)$), the predicted output vector can fill in for the state vector. Essentially, $X(k_i) \cong Y(k_i - 1) = \begin{bmatrix} y(k_i - 1 + 1|k_i - 1) \\ \vdots \\ y(k_i - 1 + N|k_i - 1) \end{bmatrix} = Fx(k_i - 1) + \phi U(k_i - 1)$. Finally applying a jerk constraint to the full control vector appears as follows in form (14) below:

$$\begin{bmatrix} \Omega & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \Omega \end{bmatrix} U(k_i) \leq \overline{J_{max}} + \begin{bmatrix} M & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & M \end{bmatrix} [Fx(k_i - 1) + \phi U(k_i - 1)] \tag{14}$$

where $\overline{J_{max}}$ is the jerk constraint expanded to a column vector of size $2N$. Feeding the jerk constraints to the solver for this system had to be done carefully. Since jerk is not a directly controlled state in this system, adding very strict jerk output constraints can cause the system to go unstable in some instances and result in a collision with the lead vehicle. Thus, despite the jerk constraint noted in Table 3, the smallest J_{max} value that could be set in (14) without causing system instability in the prescribed vehicle test scenario was 8 m/s^3. Figure 5 below shows the vehicle's state responses with the jerk constraints added.

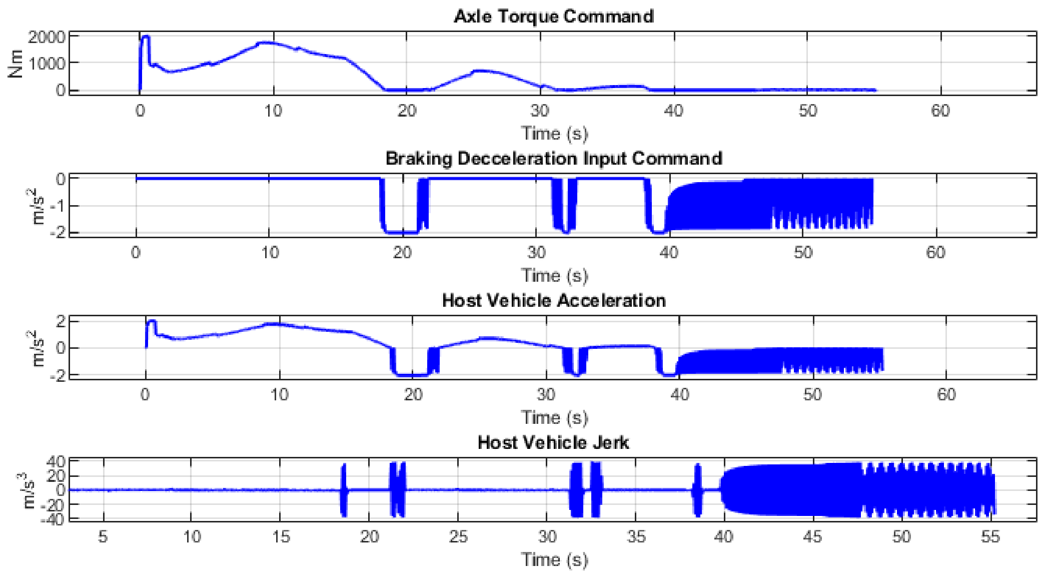

Figure 5: Vehicle jerk and acceleration states with 8 m/s^3 jerk constraint applied to unaugmented MPC at weights $q_{vh} = 0$, $q_{ah} = 20$, $q_{distError} = 100$, $q_{vrel} = 50$, $r_{Taxle} = 0.0005$, $r_{abrake} = 0.1$

The vehicle jerk graph indeed has much smaller peaks than before. The range of peaks went from ±71 m/s^3 when no jerk constraint is applied to the system to ±40 m/s^3 when it is present. While these results do show promise in limiting the jerk of the vehicle, more Q and R tuning would be required to achieve the desired performance for jerk as outlined in Table 3. Thus, it makes sense to reimplement the MPC with an augmented state-space framework where the change in control signal, and as a result vehicle jerk, can be more directly tuned for. This desire for a more methodological approach to minimizing the jerk characteristic is the motivation for the next section of this report. Moreover, with control deviation being the new input to be optimized for, the constraints on jerk can be applied as direct input constraints on *Δu* to the solver.

D. Results From MPC Implementation Based on Augmented State-Space Model

Once all of the matrices are assembled as depicted in state-space model (9), the resulting matrices for the augmented state-space system are as follows in (15) below:

$$A_{aug} = \begin{bmatrix} 1 & 0.05 & 0 & 0 & 0 & 0 \\ -0.0014 & 0.75 & 0 & 0 & 0.0003 & 0.25 \\ 0 & 0 & 1 & -0.05 & 0 & 0 \\ 0 & -0.05 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}; B_{\mathrm{aug}} = \begin{bmatrix} 0 & 0 \\ 0.0003 & 0.25 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix};$$

$$C_{aug} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \qquad (15)$$

Similar to the previous section, this set of matrices (15) is then used to determine the batch matrices from (10) and then compute the *H* and *f* matrices from (11) with current time step state vector, in this case $\xi(k_i)$, provided through feedback in Simulink at each sample step. The output constraints are applied to this new system in the same manner as previously shown in (13) with the recomputed batch matrices for the augmented system. This results in the output constraints being formulated as follows:

$$\begin{bmatrix} \phi_{aug} \\ -\phi_{aug} \end{bmatrix} \Delta U(k_i) \leq \begin{bmatrix} \overline{Y_{max}} - F_{aug}\xi(k_i) \\ -\overline{Y_{min}} + F_{aug}\xi(k_i) \end{bmatrix}$$

However, since the new control signal is Δu, the input constraints on the axle torque and braking deceleration need to be applied as shown in (16) below:

$$\begin{bmatrix} \mathrm{I} \\ -\mathrm{I} \end{bmatrix} \Delta U(k_i) \leq \begin{bmatrix} \overline{\mathrm{U}_{max}} - U(k_i - 1) \\ -\overline{\mathrm{U}_{min}} + U(k_i - 1) \end{bmatrix} \qquad (16)$$

Where *U(k_i-1)* is the horizon window optimized control vector from the previous sample step, determined with the following known signals computed as shown in form (17) below:

$$U(k_i-1)=\begin{bmatrix} u(k_i-1) \\ \vdots \\ u(k_i-1+N-1)\end{bmatrix}=\begin{bmatrix} 0 & 0 & 0 & & 0 \\ 0 & 1 & 0 & & 0 \\ 0 & 1 & 1 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 1 & 1 & & 1 \end{bmatrix}\Delta U(k_i-1)+\begin{bmatrix} u(k_i-1) \\ \vdots \\ u(k_i-1)\end{bmatrix} \tag{17}$$

Lastly, the jerk constraints are applied directly to the control deviation $\Delta u = \begin{bmatrix} \Delta T_{axle} \\ \Delta a_{brake} \end{bmatrix}$ as it's upper and lower bound vectors via the *'quadprog()'* parameters *'ub'* and *'lb'* fields respectively. Placing a jerk constraint on the braking deceleration looks like: $\frac{\Delta a_{brake}}{T_s} \leq |J_{max}|$. Since the axle torque results in acceleration from propulsion, a_{prop}, the deviation in torque would likewise result in a Δa_{prop} such that $\Delta T_{axle} = mr_w \Delta a_{prop}$. The jerk constraint applied here would look like: $\frac{\Delta a_{\text{prop}}}{T_s} \leq |J_{max}|$. Form (18) below summarizes these results:

$$\begin{bmatrix} -J_{max}T_s \\ \text{-}J_{max}mr_wT_s \end{bmatrix} \leq \begin{bmatrix} \Delta T_{axle} \\ \Delta a_{brake} \end{bmatrix} \leq \begin{bmatrix} J_{max}T_s \\ J_{max}mr_wT_s \end{bmatrix} \tag{18}$$

The lower bound and upper bound vectors from (18) are appropriately expanded to match the size of the optimized control deviation vector over the full prediction horizon, $\Delta U(k_i)$. Upon testing of certain values, a J_{max} = 3 m/s^3 was chosen as the jerk constraint value to implement in code. After some tuning, a well performing set of weights is determined to be $q_{vh} = 0$, $q_{ah} = 10$, $q_{distError} = 90$, $q_{vrel} = 110$, $r_{\Delta Taxle} = 0.005$, $r_{\Delta abrake} = 10$. Figure 6 and Figure 7 show the results from a simulation with this controller iteration.

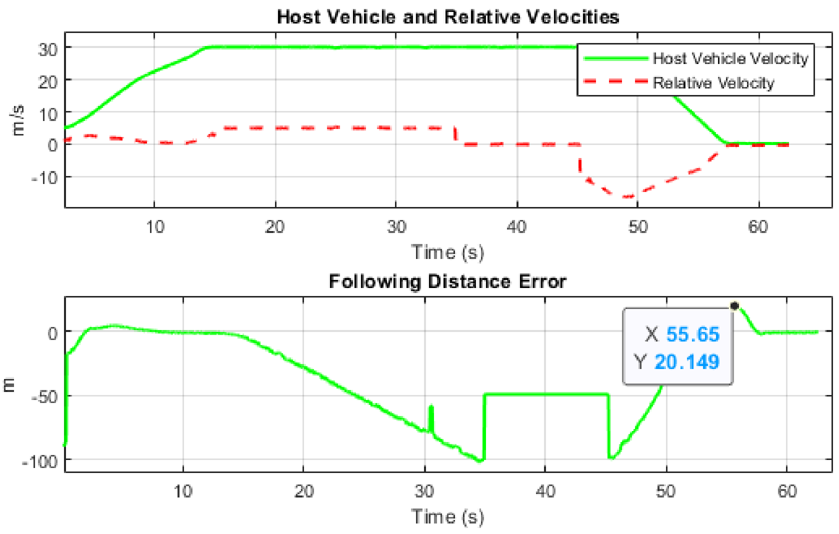

Figure 6: Augmented state-space based MPC velocity and error graphs from test run

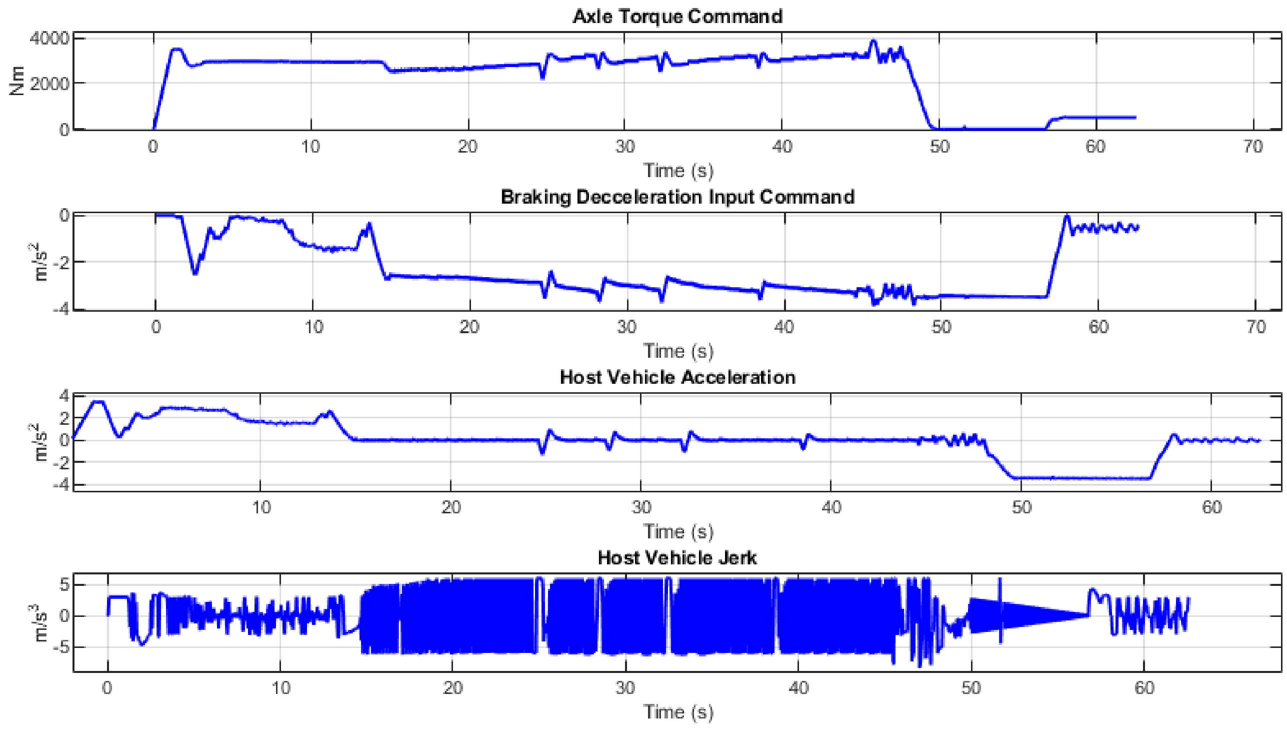

Figure 7: Augmented MPC input and acceleration and jerk graphs from test run

Figure 6 above shows that this controller managed to perform very well with the objectives of ACC, as the relative velocity is kept fairly low, meaning that the host vehicle was tracking the lead's velocity without violating the driver's own set cruise control speed limit of 30 m/s. Since the following distance error is meant to be a relaxed quantity to avoid excessive oscillations in vehicle control, this version of the controller does not exhibit bad performance in this regard from the second graph in Figure 6.

Figure 7, however, shows the jerk results, which are exceptional compared to previous controller iterations developed. Not only does the acceleration not exceed the limits of ± 3.5 m/s^2, the inputs also never violate their set limits. Moreover, all of this is accomplished while the fourth jerk graph stays relatively within the limits $\leq |5$ m/s$^3|$, which is one of the only metrics from Table 3 that could not be met with the previous controller iteration. Thus, this goes to show that the theory as expected works out, and directly constraining the control deviation can be beneficial when wanting to minimize derivative states.

Chapter 5: Conclusions and Future Work

Overall, although ADAS was only implemented on luxury vehicles in the early stages of its development, it is becoming more popularly applied and standardized on modern vehicles now. One of the most popular ADAS features is the ACC system, which provides functions such as vehicle following and autonomous driving. Hence, a need arises for understanding the working principles and the state of the art of this technology and eventually applying the principles to design an ACC system based on the MPC strategy for the University of Waterloo Alternative Fuels Team's (UWAFT) Cadillac LYRIQ vehicle.

In order to strengthen the understanding of this concept, a literature review is first conducted in this work. It is learnt that a complete ACC system often involves sensors, actuators, controller and human-machine interface. Each has its own function and collaborates to compute the appropriate throttle or braking response depending on the road conditions. The logical rules and road conditions that trigger the switching from ACC mode to CCC mode in a vehicle is also mentioned. Next, the literature review addresses the two common control structure: a one-level end-to-end controller and a hierarchical controller that consists of two levels, a high-level controller for computing the desired vehicle state and a low-level controller for converting the desired states to the proper response and commands for the vehicle. A few common control strategies are included. The PIC control has great simplicity but lacks the ability to handle non-linearity and can cause jerkiness in diving. The SMC method is effective against disturbances and uncertainties yet requires effect to tune it properly. The LQR method is good at balancing different performance factors, but it cannot enforce constraints that the vehicle may have. Lastly, the MPC control has the advantages of the LQR method, while having better abilities in predicting future disturbances, reacting towards changes and nonlinearities, and handling system constraints. Even though its computational demand is higher than the other control methods, its robustness and various advantages still make it the dominant control strategy used for ACC systems today. It is believed that the performance and function of the MPC-based ACC system can be further enhanced with future technological advancement and embedded optimization.

The modelling and simulation work carried out in this study has successfully established a detailed and physically representative framework for implementing Model Predictive Control (MPC) in an Adaptive Cruise Control (ACC) system for the 2023 Cadillac LYRIQ. Beginning with a derivation of the vehicle's longitudinal dynamics from first principles and validated parameter values, the approach systematically incorporated aerodynamic drag, actuator delays, and kinematic relationships between the host and lead vehicles to form a complete state-space representation. The model was discretized for digital control and further augmented to allow control over the rate of change of the actuator commands, enabling direct tuning of jerk behaviour.

Simulation results confirmed that the unaugmented MPC could meet basic ACC objectives when appropriately tuned, including maintaining desired speed, respecting actuator constraints, and tracking safe following distances. However, limitations were observed in jerk performance, with significant oscillations leading to reduced passenger comfort. Introducing jerk constraints into the unaugmented system improved performance but required relaxing the limits to preserve stability.

The augmented state-space MPC, which used control deviation as the optimization input, addressed these shortcomings more effectively. By directly constraining Δu, the controller achieved smooth acceleration profiles, maintained jerk within the desired $\pm 5\ m/s^3$ range, and consistently adhered to actuator and state constraints. This not only improved ride comfort but also ensured robust performance across a range of driving conditions in the simulated test scenario.

Overall, the results demonstrate that augmenting the vehicle model to include control deviations offers a more effective and methodologically sound means of shaping derivative states like jerk, while preserving tracking accuracy and respecting physical constraints. This approach provides a strong foundation for real-world deployment of an MPC-based ACC system that balances responsiveness, safety, and passenger comfort. For improving the constraints applied to the system, slack variables will be explored to allow more feasibility to the optimization problem when the states are dire. Also, solvers other than *'quadprog'* will also be researched as other solvers are known to provide faster and better convergence to a solution. Further testing related development will include Hardware in the Loop (HIL) validation of this controls code on the in vehicle controller. This will include connections with a CAN harness to simulate in vehicle communication as well.

References

[1] L.Xiao, F.Gao. (2010). A comprehensive review of the development of adaptive cruise control systems. *Vehicle System Dynamics 48:10 (2010)* 1167-1192

[2] P. Shakouri et al. (2014), Simulation Validation of Three Nonlinear Model-Based Controllers in the Adaptive Cruise Control System. *J Intell Robot Syst 80 (2015)* 207-229

[3] C. Pan. (2021). Energy-optimal adaptive cruise control strategy for electric vehicles based on model predictive control. *Energy 241 (2022)* 122793

[4] F.M. Ali, N.H. Abbas. (2024). Adaptive Cruise Control System: A Literature Survey. *Journal of Engineering 30 (2024)* 239-272

[5] L. Yu, R. Wang. (2022). Researches on Adaptive Cruise Control system: A state of the art review. *Journal of Automobile Engineering 236 (2022)* 211-240

[6] A. Petri, D.M. Petreus. (2022). Adaptive Cruise Control in Electric Vehicles with Field-Oriented Control. *Appl. Sci. 12 (2022)* 7094

[7] S. Chaturvedi, N. Kumar. (2021). Design and Implementation of an Optimized PID Controller for the Adaptive Cruise Control System. *IETE Journal of Research 69 (2023)* 7084-7091

[8] P. Shakouri et al. (2012). Designing of the Adaptive Cruise Control System-Switching Controller

[9] C. Park, H. Lee. (2017). A Study of Cruise Control System to Improve Fuel Efficiency. *International Journal of Environmental Pollution and Remediation 5 (2017)*

[10] X. Lin et al. (2017). Simplified Energy-Efficient Adaptive Cruise Control based on Model Predictive Control. *IFAC PapersOnLine 50-1 (2017)* 4794-4799

[11] (2025). Discrete State-Space. *Typhoon HIL Documentation (2025)*