# Model Predictive Control for a Basic Adaptive Cruise Control

Nikhil Nagendra (5049628),
n.hudralinagendra@student.tudelft.nl

Yen-Lin Wu (4848489),
y.wu-29@student.tudelft.nl

***Abstract*—A model predictive control (MPC) approach is implemented on a basic Adaptive Cruise Control (ACC) system. A vehicle is moving with a constant velocity and a following vehicle approaches the preceding vehicle and should maintain the same velocity. Using an MPC controller, the required stability with the specified input constraints and the target velocity for a constant preceding vehicle velocity is achieved. Through simulations in MATLAB, it is shown that the proposed MPC strategy is able to maintain a vehicle in designated constraints.**

***Index Terms*—ACC, LQR, MPC**

## I. Introduction

### A. Adaptive Cruise Control in literature

ACC is an enhancement of the conventional cruise control which is currently standardised in most modern commercialised vehicles. The purpose of a classical cruise control is to maintain longitudinal vehicle velocity by tracking the velocity as requested by the driver. In ACC, there is an additional tracking of the velocity of the vehicle ahead and adapting to it, by accelerating or braking the vehicle independently of the driver. Typically, an external sensor such as a radar is used to detect the vehicle ahead and measure relative velocity between the vehicles [1]. ACC systems consist of two subsystems: a vehicle dependent part and a vehicle independent part. The former calculates a required acceleration/deceleration profile for the vehicle. The controller part forms the dependent part which tracks the profile by actuating the throttle and brake system [2]. A schematic of the ACC system is shown in Fig. 1.

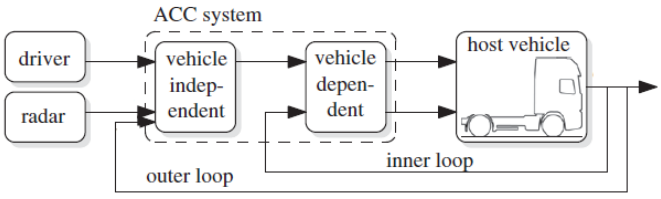


Figure 1. Schematic of ACC system [1]

The main control objective is to follow the vehicle ahead. However, there are other aspects also considered such as comfort, fuel economy and safety [1]. It should be noted that ACC is more of a comfort system than a safety system as the deceleration is limited to 3 $m/s^2$. Also, function of stop-and-go system can be integrated with ACC in order to allow the vehicle to stop behind vehicles at traffic signals or other conditions and continue again when the vehicle ahead begins to move [3]. Also, driver mental workload was reduced with the implementation of ACC. Since ACC takes control from the driver, it should resemble the driver's behaviour to a certain extent. Furthermore, constraints from subjective specific desirable behaviour should also be considered. Because of the need for constraints, classical controllers such as PID controller are insufficient. With a Linear-Quadratic Regulator (LQR) controller, it was found that the response of the system was slower compared with MPC [4]. There are 2 approaches to design controllers for linear systems with constraints: anti-windup control and MPC. Former one demonstrates adequate performance on Single-Input Single-Output (SISO) situations, whereas MPC has outperformed in the field of complex constrained multi-variable (Multi-Input Multi-Output or MIMO) control problems. Other main advantages of MPC here would be the on-line optimisation of the system, stability and feasibility guarantee with closed loop MPC, and implementation in a receding horizon manner, where the optimization problem is solved in every time step. This enables the controller to adapt to actual working conditions, i.e. traffic situations, and, as such, the controller is situation dependent. The advantage of MPC for ACC is that high control performance can be obtained since constraints such as small feedback gain for LQR controller can be exclusively dealt with and the tuned weighing parameters can be intuitively and flexibly considered as a function of time considering various driving situations [4]. In this paper, we do not consider the vehicle dynamics as this involves extracting data from engine maps and tire models to be included into the overall system which complicates the design. A detailed model with vehicle parameters is described in [5] and [6]. Also, we consider a linear model of the system for ACC. For a non-linear model, the reader is referred to [7]

### B. Vehicle Model

The host vehicle and target vehicle are modelled as in Figure 2 with a relative distance.

Since the purpose of ACC is to maintain a desired distance between host and target vehicle, a problem of the below considerations is taken,

- The distance error $\delta d$, defined as difference between the inter-vehicle distance $d$ and desired distance $d_r$, where $\delta d = d - d_r$. This error should converge to zero.

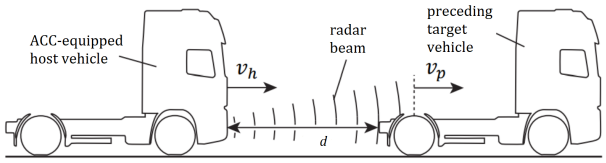


Figure 2. Example of ACC working principle. The host vehicle, driving with velocity $v_h$ and acceleration $a_h$, is equipped with ACC, which measures the preceding target vehicle, with velocity $v_p$. A radar measures the distance $x_r$ and the relative velocity $v_r = v_p - v_h$ between the vehicles. [1]

- The velocity error $\delta v$, defined as the difference between the preceding target vehicle velocity $v_p$ and host velocity $v_h$. This error also should converge to zero.
- Acceleration of the host vehicle $\dot{v}_h$, which should also converge to zero.

A vehicle dynamics model is also considered to design MPC and analyse the controller performance. The longitudinal dynamics of the host vehicle is given as,

$$m\dot{v}_h = ma_f - r_{travel}$$

Here, $m$ is the mass of the vehicle, $a_f$ is the traction force of the host vehicle converted to acceleration and $r_{travel}$ is the travel resistance consisting of several factors. In general, the actuation dynamics can be described as an ordinary differential equation as below,

$$\begin{aligned} \dot{v}_h &= f_{act}(x_f, u) \\ a_f &= h_{act}(v_h) \end{aligned} \tag{1}$$

where $v_h \in \mathbb{R}^{n_f}$, $u \in \mathbb{R}$ are the states and the input to actuation system respectively. $u$ is the acceleration command calculated by the ACC controller. The output of the system is y = $a_f$. It is to be noted that since we are taking a simple model, we neglect $r_{travel}$ as it contains several factors which are vehicle specific.

For the plant model of the ACC system, two primary state variables are taken, $\delta d$ and $\delta v$ as described earlier. The variable $d_r$ defined in $\delta d$ is given based on the constant time headway given as

$$d_r = T_{hw} v_h + d_0 \tag{2}$$

where $T_{hw}$ is the constant time headway (the time needed for the host vehicle to reach the current position of the target vehicle) and $d_0$ is the stopping distance for safety margin. Here, we do not consider the stop-and-go function where the vehicle stops completely if the target vehicle stops. Hence we take $d_0$ as zero.

Now we define the state variable of the plant as,

$$x = \begin{bmatrix} x_1 & x_2 & x_3^T \end{bmatrix}$$

with $x_1 = \delta d$, $x_2 = \delta v$ and $x_3 = \dot{v}_h$.
The state-space model can now be formulated as,

$$\begin{aligned} \dot{v}_h &= A_f(t) v_h + B_f(t) u \\ a_f &= C_f v_h \end{aligned} \tag{3}$$

where the state $\dot{v}_h \in \mathbb{R}$, i.e. $n_f = 1$ and

$$\begin{aligned} A_f(t) &= -\frac{1}{T_{eng}} \\ B_f(t) &= -\frac{K_{eng}(t)}{T_{eng}} \end{aligned} \tag{4}$$

$$C_f = 1$$

$T_{eng}$ is given as the time constant of acceleration using engine; $K_{eng}(t)$ is the steady-state gains dependent on time. In the system 1, the dynamics is separated into acceleration and deceleration side and simply modeled as first-order delay system. Therefore, the overall simplified linear state-space model with the three states can be written as,

$$\begin{aligned} \dot{x} &= A(t)x + B(t)u \\ y &= Cx \end{aligned} \tag{5}$$

where $x \in \mathbb{R}^3$ and

$$A(t) = \begin{bmatrix} 0 & 1 & -T_{hw} \\ 0 & 0 & -1 \\ 0 & 0 & A_f(t) \end{bmatrix}, \quad B(t) = \begin{bmatrix} 0 \\ 0 \\ B_f(t) \end{bmatrix}$$

In this model, the rank of $A = 2$, which implies that the defined system is not controllable. In this case, not all inputs transfers all states to zero states or any other state. Since only the host velocity can directly be controlled, controllability of the system is not of much importance. The other 1 state variable, $\delta v$, is involved with $v_h$ and $v_p$.

The model (5) will be used for our MPC problem. For a non-linear system, it is necessary to linearize the system around the equilibrium. However, since we have already considered a linear system, we directly discretize the system from continuous time to discrete time. In MATLAB, this is done using the `c2d` command. Hence, the below discretized model is obtained.

$$\begin{aligned} x_{t+1} &= \Pi(t)x_t + \Gamma(t)u_t \\ y_t &= C_d x_t \end{aligned} \tag{6}$$

$\Pi(t) = (e^{A(t).T_s})$, $\Gamma(t) = \int_0^{T_s} e^{A(t)s} ds \cdot B(t)$ and $C_d = C$ where $T_s$ is the sampling time.

## II. Model Predictive Control Design

In this section, a model predictive control method is designed to maintain a constant distance between the target and host vehicle. Different MPC design methods were considered. We use the method of regulation of system using state-based MPC. To solve the MPC problem at every iteration, we define a general cost function as given in [8]. This cost function is defined as follows,

$$V_N(x_0, u) = \sum_{k=0}^{N-1} \big\{ l(x(k), u(k)\} + V_f\big(x(N)\big) \tag{7}$$

$$\text{s.t. } u \in \mathbb{U}, x \in \mathbb{X}$$

Here, $\delta d$, $\delta v$ and $\dot{v}_h$ have to be controlled to converge to the origin for a given control input command $u(k)$. The states considered have to be constrained in order to meet the required distance and velocity between the vehicles. The ACC system is a comfort system and hence the accelerations are limited to $\dot{v}_{h_{min}} = u_{min} = -3.0\ m/s^2$. The maximum acceleration however depends on the $v_p$. Due to the different nature of engines between the vehicles, it may not be possible to achieve the same acceleration. Hence, the maximum acceleration $a_{h_{max}}$ is bounded. The relation between $\dot{v}_h$ and $v_p$ is chosen linear due to the MPC problem taken. Therefore, $\dot{v}_{h_{max}} = 3(1 - 0.025 \cdot v_h)\ m/s^2$ [3]. It is assumed that the $v_h = 40\ m/s$ which results in $\dot{v}_{h_{max}} = 0$. Also, a constraint is added on the rate of change of acceleration, $|\Delta\dot{v}| \leq 5\ m/s^3$, to minimize the jerk. Finally, the minimum $d$ should be greater than $5\ m$ in order to avoid collision with the preceding target vehicle. (This constraint may be subject to local laws that may apply.) State boundaries are defined as follows,

- Maximum and minimum $v_h$: Here, only longitudinal movement of the host vehicle is considered.
- Maximum and minimum $v_p$: Here, only longitudinal movement of the preceding target vehicle is considered as well. Also, it is assumed that the maximum $v_h$ is the same as the $v_p$ and both are moving in the same direction.
- Since we do not consider stop-and-go function, we set the lower bound of $v_p$ to $15\ m/s$.
- Maximum $d$ is the maximum range of the radar.

$$m = \begin{bmatrix} 0 \\ 0 \\ 15 \end{bmatrix} \leq \begin{bmatrix} \delta d \\ \delta v \\ v_h \end{bmatrix} \leq \begin{bmatrix} 2 \\ 2.5 \\ 40 \end{bmatrix} = M \tag{8}$$

where $v_h$ is obtained from $\dot{v}_h$.

Constraints are also put on the controller unit. The acceleration of the vehicle should not exceed $5\ m/s^2$ and the deceleration should be greater than $-3\ m/s^2$. The input bound could then be written as follows:

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} u(k) \leq \begin{bmatrix} 5 \\ 3 \end{bmatrix} \tag{9}$$

Note: The state constraints (equation (8)) are used for calculation of the hyperrectangle in III. For simulation in IV, $v_p$ is taken as a constant. Therefore, the only constraints taken are (6) and (9), while the state constraints are ignored.

The MPC optimisation problem to be solved for each iteration is represented by the equation (7) and hence is given as,

$$\begin{aligned} l\big(x(k), u(k)\big) &= x(k)^T Q x(k) + u(k)^T R u(k) \\ V_f\big(x(N)\big) &= x(N)^T P x(N) \end{aligned} \tag{10}$$

where $Q = I$ , $R = 1$ and $P$ is the solution to the Discrete Algebraic Riccati Equation (DARE). $P$, $Q$ and $R$ are positive definite in this case. The values of $R$ and $Q$ were obtained in an iterative manner.

## III. Asymptotic Stability

In this section, we show that the MPC controller strategy proposed in section II asymptotically stabilizes the closed loop system given that the system initial conditions are sufficiently close to the origin. Asymptotic (Exponential) Stability:

$$\begin{aligned} l(x,u) &= \frac{1}{2}(x^T Q x + u^T R u) \geq \frac{1}{2} x^T Q x \geq \frac{1}{2}\lambda_{min}(Q)|x|^2 \\ V_f(x) &= \frac{1}{2} x^T P x \leq \frac{1}{2}\lambda_{max}(P)|x|^2 \end{aligned} \tag{11}$$

### A. Auxiliary results

We need to verify the following 2 auxiliary results:
1) $\mathbb{X}_f \subseteq \{x \in \mathbb{X} \mid Kx \in \mathbb{U}\}$
2) $x \in \mathbb{X}_f \Rightarrow A_K x \in \mathbb{X}_f$

To prove 1), we assume $x \in \mathbb{X}_f$ holds. We have

$$x^T P x \leq 1$$

Note that equation (8) that describes a 3-dimensional constraint could be written as follows

$$\mathbb{X} = \{x \in \mathbb{R}^3 \mid m \leq x \leq M\} \tag{12}$$

As $P$ is symmetric and positive definite, it is orthogonally diagonalizable, i.e. $P = SDS^T$, where $D$ is a matrix containing the eigenvalues of $P$ on the diagonal and $S$ is an orthogonal matrix containing the eigenvectors of $P$, with $S^{-1} = S^T$. Thus, the hyperrectangle can be described by

$$x^T SDS^T x \leq R$$

By introducing coordinate transformation $x = Sy$, the hyperrectangle can be written as

$$y^T D y \leq 1$$

Same could be applied to equation (12):

$$\mathbb{X} = \{y \in \mathbb{R}^3 \mid m \leq Sy \leq M\} \tag{13}$$

To see if the hyperrectangle is contained in $\mathbb{X}$, we could also check if its vertices are within $\mathbb{X}$.

As shown in Figure 3, $\mathbb{X}$ is the red diamond shaped region. We need to prove that the pink points ($\mathbb{X}_f$) belongs to $\mathbb{X}$ in III-D, i.e. $\mathbb{X}_f \subseteq \mathbb{X}$.

We referenced the code from `alexberndt` [1] for the calculation of hyperrectangle.

### B. Assumption 2.2 [8]

This assumption is to check the continuity of the system and cost. The assumption here is satisfied since the $l(x,u)$ and $V_f(x)$ are continuous and positive definite functions (as they are linear) as defined in Section II. Also, since linearity of the system implies that it is continuous and the equilibrium point is $(0,0)$. Thus the assumption is satisfied.

[1] https://github.com/alexberndt/Quadrotor-Balancing-Pendulum-MPC

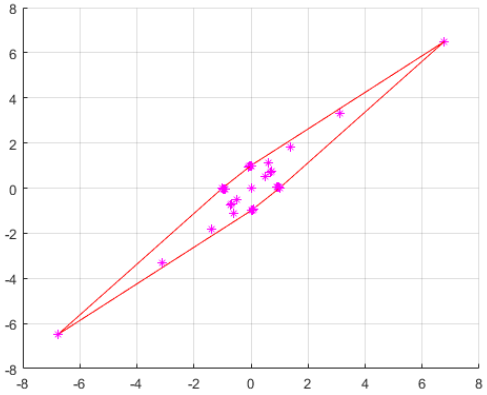

Figure 3. Hypterractangle

### C. Assumption 2.3 [8]

To validate this assumption, the set $\mathbb{Z}$ should be closed and the set $\mathbb{X}_f \subseteq \mathbb{X}$ should be compact. This is valid as both the sets $u \in \mathbb{X}$ and $u \in \mathbb{U}$ are closed, compact and contain the origin as described in II.

### D. Assumption 2.14 [8]

This assumption is to show that the optimal terminal cost function $V_N^0(x)$ is a valid Lyapunov function. In addition to assumptions 2.2 and 2.3 being satisfied, it is needed to prove assumption 2.14 by finding the maximum invariant constraint admissible set $\mathbb{X}_f$ and the set $\mathcal{X}_N$.

The lower bound on $V_N^0(x)$ is easily obtained since $V_N^0(x) \geq l(x, K_N(X))$ for all $X \in \mathcal{X}_N$. $\mathbb{X}_f$ is the set that ensures that the optimal control input will be same as the unconstrained control input defined by the infinite-horizon LQR problem.

The upper bound on $V_N^0(x)$ can be established if $\mathbb{X}_f$ contains the origin in its interior and $V_N^0(x)$ is locally bounded.

We make use of a modification of the Algorithm 3.2 of [9] to obtain an estimated set $\mathbb{X}_f$ which follows the control input and state constraints. A state and input admissible invariant set which will hold all the states to ensure the unconstrained LQR control law will keep the state within. Since there are only 3 states in this case, we can plot them on 2D and 3D in order to show the set $\mathbb{X}_f$ as seen in Figure (3).

## IV. Numerical Simulation

The prediction horizon $p = 20$ and control horizon $c = 1$ are chosen to be as small as possible. Initial host acceleration $\dot{v}_0$ is set to 15 $m^2/s$. The parameters for actuation model is given in Table I. To simulate the MPC controller, we use `quadprog` for Quadratic Programming (QP) in MATLAB as the cost function is quadratic and the constraints are linear. The QP solver available are active-set and interior-point methods. We use the MATLAB built-in solver `interior-point-convex`.

The simulation was performed on an i7-processor laptop with 16GB RAM configuration.

Table I
Parameters of actuation model

| Notation | Value | Unit |
|---|---|---|
| $T_s$ | 0.050 | $s$ |
| $T_{hw}$ | 1.300 | $s$ |
| $T_{eng}$ | 0.460 | $s$ |
| $K_{eng}$ | 0.732 | $(-)$ |

### A. MPC Output

As seen in the figure (4), the MPC controller is able to minimize $\delta d$. Negative $\delta d$ does not imply that the vehicle crashes into the other, however, it implies that the difference between maintained distance (equal to specified limit of say $4m$) and actual distance is zero. $v_p$ maintains a constant velocity of 15 $m/s$. The $\delta v$ and $\dot{v}_h$ also converges to zero in a short period of time. A negative $\delta v$ implies that the both the cars are moving towards each other with zero velocity error implying that they travel with the same velocity after some time.

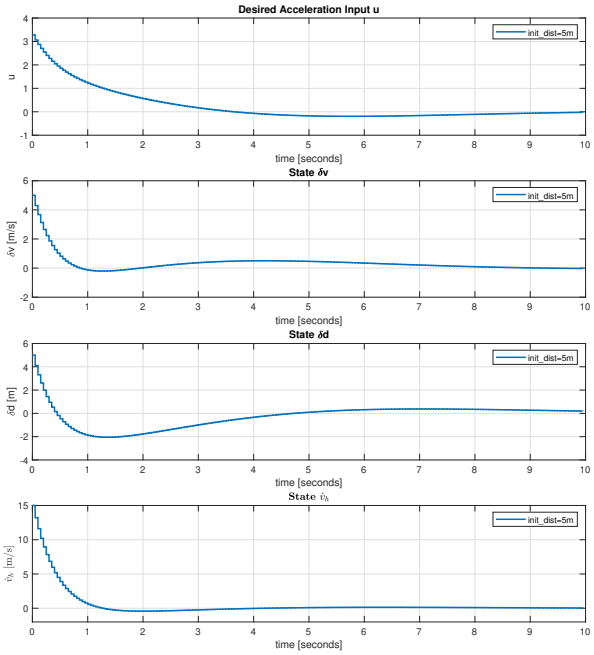


Figure 4. MPC Result

### B. Comparison between LQR and MPC output

A comparison between LQR and MPC controller outputs can be seen in figure (5). The LQR controller output to the plant is at a higher rate compared to the MPC counterpart. Both $\delta d$ and $\delta v$ stabilize much faster as compared to MPC. As seen, the LQR controller is able to perform slightly better that MPC controller. However, the input acceleration rate is much higher than MPC which causes an uncomfortable situation for the driver and passengers as explained in I.

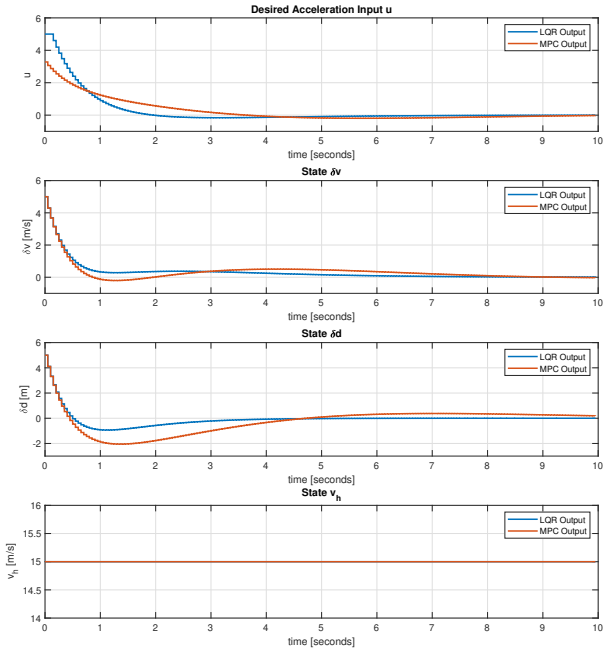


Figure 5. LQR and MPC comparison

## C. Varying Horizon $N$

Figure (6) shows the different prediction horizons in MPC. For a low prediction horizon of $N = 5$ and $N = 10$, the controller performs poorly and does not stabilize. For a horizon of $N = 20$ and above, the controller is able to reach stability. However, choosing a very large horizon can increase computation time so in our case, a feasible prediction horizon of $N = 20$ was chosen. and implemented for other cases.

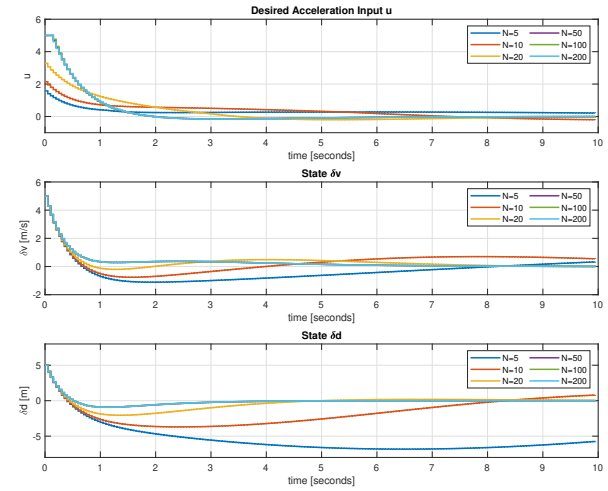


Figure 6. MPC results with varying N

## D. Varying State Weight $Q$

Figure (7) shows the different results obtained by varying the state weight $Q$. It can be seen that for a weight $Q = I$, the controller output is able to achieve zero error in a comfortable way; whereas for $Q = 0.1 * I$, the controller is more aggressive which, although satisfies the constraints, is not preferred for a comfortable drive.

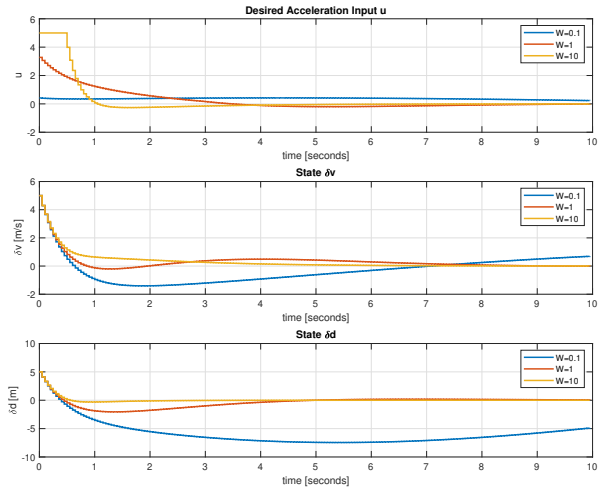


Figure 7. MPC results with varying Q

## E. Varying State Weight $R$

The same holds for varying the weight $R$ as well. As seen in figure (8), the most optimal controller would be with $R = 1$.

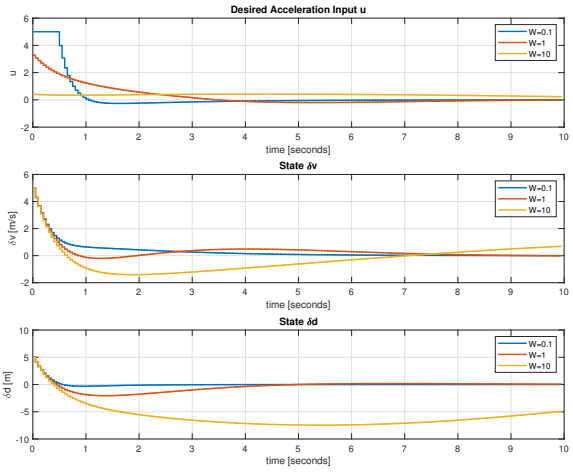


Figure 8. MPC results with varying R

Therefore, with a weight $Q = I$ and $R = 1$, we can achieve a controller suitable for our specific use case.

## V. Conclusions

In this paper, we implement a simple MPC controller for a basic version of ACC. The controller is able to achieve the required stability and ensure that the distance between the car is minimized to a short time. In our case of comparing LQR and MPC controller, the LQR performs better as it is able to reduce the error in a shorter time as compared to MPC. However, this is not preferable since it can cause a discomfort for the driver and passenger. An MPC controller would hence be preferred in this case. With the implementation of state constraints as well, the controller would be able to achieve the required stability for a variety of cases, especially where the distance between the vehicles is large. This destabilizes the plant and the controller is unable to achieve the stability. Hence, in this case where the difference in distance between the host and preceding vehicle is low, the controller performs satisfactorily.

The controller is also tested for different prediction horizons and performs well at a horizon of $N = 20$, beyond which no significant improvement is observed. An increase in the horizon would only increase computation time, which is critical in a case such as ACC.

The implemented MPC here is without state constraints (8). For future work, state constraints can be included and tested with different initial distances between vehicles. This improves the robustness of the controller and brings it closer to a practical scenario. Also, the controller can be implemented with a varying $v_p$ and verify that when the preceding car accelerates, the controller is able to adapt and maintain the same velocity which is the ultimate goal of ACC.

One of the drawbacks of MPC is that simulation is performed online and an accurate model or representation of the model is needed. In our case, the model is moderately accurate for the task and initial conditions given. Due to the simplicity of the model, the optimisation can be done fast with satisfactory results.

## References


[1] G.J.L. Naus, J. Ploeg, M.J.G. Molengraft, van de, W.P.M.H. Heemels, and M. Steinbuch. Design and implementation of parameterized adaptive cruise control : an explicit model predictive control approach. *Control Engineering Practice*, 18(8):882–892, 2010.

[2] Seungwuk Moon, Ilki Moon, and Kyongsu Yi. Design, tuning, and evaluation of a full-range adaptive cruise control system with collision avoidance. *Control Engineering Practice*, 17:442–455, 04 2009.

[3] R.A.P.M. van den Bleek. Design of a hybrid adaptive cruise control stop-&go system. Master's thesis, Technische Universiteit Eindhoven, 5612 AZ Eindhoven, 2007.

[4] Taku Takahama and Daisuke Akasaka. Model predictive control approach to design practical adaptive cruise control for traffic jam. *International Journal of Automotive Engineering*, 9(3):99–104, 2018.

[5] V. L. Bageshwar, W. L. Garrard, and R. Rajamani. Model predictive control of transitional maneuvers for adaptive cruise control vehicles. *IEEE Transactions on Vehicular Technology*, 53(5):1573–1585, 2004.

[6] E. Kural and B. Aksun Güvenç. Model predictive adaptive cruise control. In *2010 IEEE International Conference on Systems, Man and Cybernetics*, pages 1455–1461, 2010.

[7] P. Shakouri and A. Ordys. Application of the state-dependent nonlinear model predictive control in adaptive cruise control system. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 686–691, 2011.

[8] J. Rawlings, D.Q. Mayne, and Moritz Diehl. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 01 2017.

[9] E. G. Gilbert and K. T. Tan. Linear systems with state and control constraints: the theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36, no. 9:1008–1020, Sept. 1991.


## Nomenclature

| | |
|---|---|
| $\delta d$ | Distance error |
| $\delta v$ | Velocity error |
| $\dot{v}_h$ | Host vehicle acceleration |
| $a_f$ | Host vehicle traction force converted to acceleration |
| $a_{h_{max}}$ | Maximum acceleration |
| $d$ | Inter-vehicle distance |
| $d_0$ | Stopping distance |
| $d_r$ | Desired distance |
| $K_{eng}$ | Steady-state gain |
| $m$ | Vehicle mass |
| $r_{travel}$ | Travel resistance |
| $T_{eng}$ | Time constant of acceleration using engine |
| $T_{hw}$ | Constant time headway |
| $u$ | Control input command |
| $v_h$ | Host vehicle velocity |
| $v_p$ | Preceding vehicle velocity |