ELEN90095 AI4R
Real-Time Iteration MPC for Autonomous Vehicle Control
November 2024

Jordan Bierbrier

Contents

1 State Definition

The vehicle dynamics for the Model Predictive Control (MPC) implementation are based on a kinematic bicycle model with four states:

- p — Progress along the road [m]: the distance travelled along the reference path.
- d — Orthogonal distance to the road [m]: the lateral deviation from the reference path.
- μ — Heading angle relative to the path [rad]: the difference between the vehicle's orientation and the path tangent direction.
- v — Velocity at the rear wheel [m/s]: the longitudinal speed of the vehicle measured at the rear axle.

These states are used to describe the system dynamics and are controlled via two inputs: the steering angle δ [rad] and the drive command F_{cmd} (normalized [-100, 100]%). The continuous-time dynamics are modeled as:

$$\begin{aligned} \dot{p} &= v\cos(\mu)\frac{1}{1+d\kappa(p)} \\ \dot{d} &= v\sin(\mu) \\ \dot{\mu} &= \frac{v\tan(\delta)}{l_r} - \frac{\kappa(p)v\cos(\mu)}{1+d\kappa(p)} \\ \dot{v} &= \frac{1}{m}F_{\text{cmd}}C_m - \frac{1}{m}C_d v^2\,\text{sign}(v) \end{aligned} \tag{1}$$

where:

- $\kappa(p)$ is the curvature of the road at progress p,
- l_r is the distance from the rear axle to the center of mass,
- m is the mass of the vehicle,
- C_m and C_d are motor and drag coefficients, respectively.

2 Real-Time Iteration MPC

2.1 Derivation

The path following task can be formulated as nonlinear optimization problem as

$$\begin{aligned} \min_{\mathbf{s}_k,\mathbf{a}_k} \quad & \frac{1}{2}\sum_{k=0}^{N-1} \|\mathbf{s}_k - \bar{\mathbf{s}}_k\|^2_{\mathbf{Q}} + \|\mathbf{a}_k - \bar{\mathbf{a}}_k\|^2_{\mathbf{R}} \\ \text{s.t.} \quad & \mathbf{s}_{k+1} = f_d(\mathbf{s}_k, \mathbf{a}_k), \quad k = 0, ..., (N-1) \\ & \mathbf{A}_k \begin{bmatrix} \mathbf{s}_k \\ \mathbf{a}_k \end{bmatrix} \leq \mathbf{b}_k, \quad k = 0, ..., (N-1), \end{aligned} \tag{2}$$

where the objective is to minimize the difference between the states, $\mathbf{s}_k \in \mathbb{R}^4$, and state references, $\bar{\mathbf{s}}_k \in \mathbb{R}^4$, and the difference between the actions, $\mathbf{a}_k \in \mathbb{R}^2$, and action references, $\bar{\mathbf{a}}_k \in \mathbb{R}^2$, over a time horizon, N. $\mathbf{Q} \in \mathbb{R}^{4\times4}$ and $\mathbf{R} \in \mathbb{R}^{2\times2}$ represent the weighting of the state and action differences, respectively. Additionally, the optimization is subject to dynamic constraints through f_d, and state and action constraints (more on this later), over the entire horizon. The nonlinear optimization problem, 2, can be solved using real-time iteration (RTI) MPC [1]. For this, 2 must be translated into a convex optimization problem with the following structure

$$\begin{aligned}
\min_{\mathbf{s}_k,\mathbf{a}_k} \quad & \frac{1}{2}\sum_{k=0}^{N-1} \|\mathbf{s}_k - \bar{\mathbf{s}}_k\|^2_{\mathbf{Q}} + \|\mathbf{a}_k - \bar{\mathbf{a}}_k\|^2_{\mathbf{R}} \\
\text{s.t.} \quad & \mathbf{s}_{k+1} = \mathbf{A}_{d,k}\mathbf{s}_k + \mathbf{B}_{d,k}\mathbf{a}_k + \mathbf{g}_{d,k}, \qquad k = 0, ..., (N-1) \\
& \mathbf{s}_0 = \mathbf{s}(t_{\text{current}}), \\
& \mathbf{l}_a \leq \mathbf{a}_k \leq \mathbf{u}_a, \qquad\qquad k = 0, ..., (N-1),
\end{aligned} \tag{3}$$

where the objective function remains the same, however the constraints have been modified. Specifically, the dynamic constraint now consists of a linear approximation using $\mathbf{A}_{d,k} \in \mathbb{R}^{4\times4}$, $\mathbf{B}_{d,k} \in \mathbb{R}^{4\times2}$, and $\mathbf{g}_{d,k} \in \mathbb{R}^4$ around the linearization point, $\mathbf{s}_k$. Next, the initial state constraint can be seen explicitly in 3 through the equality constraint on $\mathbf{s}_0$. Similarly, the action constraints are seen more explicitly as being lower and upper bounded at each time step.

To solve 3 using a standard solver (i.e. OSQP), we must further translate 3 into a condensed quadratic program (QP) form with a single optimization variable, $\mathbf{x}$, as

$$\begin{aligned}
\min_{\mathbf{x}} \quad & \frac{1}{2}\mathbf{x}^T\mathbf{P}\mathbf{x} + \mathbf{q}^T\mathbf{x} \\
\text{s.t.} \quad & \mathbf{l} \leq \mathbf{A}\mathbf{x} \leq \mathbf{u}.
\end{aligned} \tag{4}$$

Several steps must be followed to translate 3 into 4. These steps are shown below.

First, we must first define a new optimization variable as

$$\mathbf{x} := \begin{bmatrix} \mathbf{s}_0 \\ \mathbf{a}_0 \\ \mathbf{s}_1 \\ \mathbf{a}_1 \\ \vdots \\ \mathbf{s}_{N-1} \\ \mathbf{a}_{N-1} \\ \mathbf{s}_N \end{bmatrix}.$$

Next, the objective function must be expressed in terms of the newly defined optimization variable, $\mathbf{x} \in \mathbb{R}^{6N+4}$. The objective function from 3,

$$\frac{1}{2}\sum_{k=0}^{N-1} \|\mathbf{s}_k - \bar{\mathbf{s}}_k\|^2_{\mathbf{Q}} + \|\mathbf{a}_k - \bar{\mathbf{a}}_k\|^2_{\mathbf{R}},$$

$$= \frac{1}{2}\sum_{k=0}^{N-1} (\mathbf{s}_k - \bar{\mathbf{s}}_k)^T\mathbf{Q}(\mathbf{s}_k - \bar{\mathbf{s}}_k) + (\mathbf{a}_k - \bar{\mathbf{a}}_k)^T\mathbf{R}(\mathbf{a}_k - \bar{\mathbf{a}}_k),$$

can be expanded to

$$\frac{1}{2}\sum_{k=0}^{N-1} \mathbf{s}_k^T\mathbf{Q}\mathbf{s}_k - 2\bar{\mathbf{s}}_k^T\mathbf{Q}\mathbf{s}_k + \underbrace{\bar{\mathbf{s}}_k^T\mathbf{Q}\bar{\mathbf{s}}_k}_{(*)} + \mathbf{a}_k^T\mathbf{R}\mathbf{a}_k - 2\bar{\mathbf{a}}_k^T\mathbf{R}\mathbf{a}_k + \underbrace{\bar{\mathbf{a}}_k^T\mathbf{R}\bar{\mathbf{a}}_k}_{(**)}.$$

Since the terms $(*)$ and $(**)$ do not depend on the new optimization variable $\mathbf{x}$, they can be dropped from the objective function. This results in the updated objective function as

$$\frac{1}{2}\sum_{k=0}^{N-1} \underbrace{\mathbf{s}_k^T\mathbf{Q}\mathbf{s}_k + \mathbf{a}_k^T\mathbf{R}\mathbf{a}_k}_{(\dagger)} \underbrace{-2\bar{\mathbf{s}}_k^T\mathbf{Q}\mathbf{s}_k - 2\bar{\mathbf{a}}_k^T\mathbf{R}\mathbf{a}_k}_{(\ddagger)},$$

which has a quadratic component and a linear components, denoted by (†) and (‡), respectively. These terms can be written in matrix form and represent the updated objective function

$$f(\mathbf{x}) := \frac{1}{2}\mathbf{x}^T\mathbf{P}\mathbf{x} + \mathbf{q}^T\mathbf{x},$$

with

$$\mathbf{P} = \begin{bmatrix} \mathbf{Q} & \mathbf{0} & \mathbf{0} & \dots & & & \mathbf{0} \\ \mathbf{0} & \mathbf{R} & \mathbf{0} & \dots & & & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Q} & & & & \vdots \\ \vdots & \vdots & & \ddots & & & \\ \mathbf{0} & \dots & & & \mathbf{Q} & & \\ \mathbf{0} & \dots & & & & \mathbf{R} & \\ \mathbf{0} & \dots & & & & & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{(6N+4)\times(6N+4)},$$

$$\mathbf{q} = \begin{bmatrix} -\mathbf{Q}\bar{\mathbf{s}}_0 \\ -\mathbf{R}\bar{\mathbf{a}}_0 \\ -\mathbf{Q}\bar{\mathbf{s}}_1 \\ -\mathbf{R}\bar{\mathbf{a}}_1 \\ \vdots \\ -\mathbf{Q}\bar{\mathbf{s}}_{N-1} \\ -\mathbf{R}\bar{\mathbf{a}}_{N-1} \\ \mathbf{0} \end{bmatrix} \in \mathbb{R}^{6N+4}.$$

Next, the constraints of 3 must be expressed in terms of $\mathbf{x}$ and in the form of 4. First, the linear dynamics constraint can be rearranged to

$$\mathbf{A}_{d,k}\mathbf{s}_k + \mathbf{B}_{d,k}\mathbf{a}_k - \mathbf{s}_{k+1} = -\mathbf{g}_{d,k}, \quad k = 0, ..., (N-1)$$

and written in terms of $\mathbf{x}$ in matrix form as

$$\tilde{\mathbf{g}} \leq \tilde{\mathbf{A}}_d\mathbf{x} \leq \tilde{\mathbf{g}}, \tag{5}$$

where

$$\tilde{\mathbf{A}}_d = \begin{bmatrix} \mathbf{A}_{d,0} & \mathbf{B}_{d,0} & -\mathbf{I}_{4\times4} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{d,1} & \mathbf{B}_{d,1} & -\mathbf{I}_{4\times4} & \mathbf{0} & \dots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & & \ddots & \ddots & & \\ \vdots & \vdots & \vdots & & & \ddots & \ddots & \\ \mathbf{0} & \dots & & & & \mathbf{A}_{d,N-1} & \mathbf{B}_{d,N-1} & -\mathbf{I}_{4\times4} \end{bmatrix} \in \mathbb{R}^{4N\times(6N+4)},$$

$$\tilde{\mathbf{g}} = \begin{bmatrix} -\mathbf{g}_{d,0} \\ -\mathbf{g}_{d,1} \\ \vdots \\ -\mathbf{g}_{d,N-1} \end{bmatrix} \in \mathbb{R}^{4N}.$$

The initial state constraint of 3 takes the form

$$\tilde{\mathbf{s}} \leq \tilde{\mathbf{I}}\mathbf{x} \leq \tilde{\mathbf{s}}, \tag{6}$$

where

$$\tilde{\mathbf{I}} = \begin{bmatrix} \mathbf{I}_{4\times 4} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots \end{bmatrix} \in \mathbb{R}^{4\times(6N+4)},$$

$$\tilde{\mathbf{s}} = \mathbf{s}(t_{\text{current}}) \in \mathbb{R}^4.$$

Lastly, the action space constraint from 3 can be expressed as

$$\tilde{\mathbf{l}}_a \leq \tilde{\mathbf{I}}_a\mathbf{x} \leq \tilde{\mathbf{u}}_a, \tag{7}$$

where

$$\tilde{\mathbf{I}}_a = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{2\times 2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{2\times 2} & \mathbf{0} & \dots & & \vdots \\ \vdots & \vdots & & & \ddots & \ddots & & \\ \mathbf{0} & \dots & & & & \mathbf{0} & \mathbf{I}_{2\times 2} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{2N\times(6N+4)},$$

$$\tilde{\mathbf{l}}_a = \begin{bmatrix} \mathbf{l}_a \\ \mathbf{l}_a \\ \vdots \\ \mathbf{l}_a \end{bmatrix} \in \mathbb{R}^{2N},$$

$$\tilde{\mathbf{u}}_a = \begin{bmatrix} \mathbf{u}_a \\ \mathbf{u}_a \\ \vdots \\ \mathbf{u}_a \end{bmatrix} \in \mathbb{R}^{2N}.$$

Constraints 5, 6 and 7 can be stacked resulting in the final constraint

$$\underbrace{\begin{bmatrix} \tilde{\mathbf{g}} \\ \tilde{\mathbf{s}} \\ \tilde{\mathbf{l}}_a \end{bmatrix}}_{\mathbf{l}} \leq \underbrace{\begin{bmatrix} \tilde{\mathbf{A}}_d \\ \tilde{\mathbf{I}} \\ \tilde{\mathbf{I}}_a \end{bmatrix}}_{\mathbf{A}} \mathbf{x} \leq \underbrace{\begin{bmatrix} \tilde{\mathbf{g}} \\ \tilde{\mathbf{s}} \\ \tilde{\mathbf{u}}_a \end{bmatrix}}_{\mathbf{u}},$$

where $\mathbf{l}, \mathbf{u} \in \mathbb{R}^{(4N+4+2N)}$ and $\mathbf{A} \in \mathbb{R}^{(4N+4+2N)\times(6N+4)}$.

2.2 Control Algorithm

Algorithm 1 is used for trajectory following. It includes the previously mentioned steps along with solving with the optimization problem. It should be noted that Algorithm 1 does not show all the specific details (i.e. setting up action constraints), however, it highlights the major components.

Algorithm 1 RTI-MPC Simulation Loop

$t \leftarrow 0$
$\mathbf{s}_{pred}, \mathbf{a}_{pred} \leftarrow$ `initialize_simulation`$(N, T_s, \mathbf{s_0})$
$\mathbf{s}_{ref}, \mathbf{a}_{ref} \leftarrow$ `reference_path`(t, N, T_s)
while $(\mathbf{s}_{ref}, \mathbf{a}_{ref})$ **do**
  $\mathbf{s_0} \leftarrow$ `measure_current_state`()
  $\mathbf{s}_{lin} \leftarrow \mathbf{s}_{pred}$
  $\mathbf{s}_{lin}[0] \leftarrow \mathbf{s_0}$
  $\mathbf{a}_{lin} \leftarrow \mathbf{a}_{pred}$
  $\kappa_{lin} \leftarrow$ `get_curvature`$(\mathbf{s}_{lin})$
  $\tilde{\mathbf{A}}_d, \tilde{\mathbf{g}} \leftarrow$ `compute_linearization`$(\mathbf{s}_{lin}, \mathbf{a}_{lin}, \kappa_{lin})$
  $\mathbf{s}_{opt}, \mathbf{a}_{opt} \leftarrow$ `solve_mpc`$(\tilde{\mathbf{A}}_d, \tilde{\mathbf{g}}, \mathbf{s}_{ref}, \mathbf{a}_{ref})$
  `apply_control`$(\mathbf{a}_{opt}[0])$
  $\mathbf{s}_{pred}, \mathbf{a}_{pred} \leftarrow$ `update_predictions`$(\mathbf{s}_{opt}, \mathbf{a}_{opt})$
  $t \leftarrow t + T_s$
  $\mathbf{s}_{ref}, \mathbf{a}_{ref} \leftarrow$ `reference_path`(t, N, T_s)
end while

2.3 Reference Trajectories

As it can be seen in equation 2, the objective is to remain as close to the reference states and actions as possible. Therefore, reference states and actions must be passed to the optimization problem to solve the task. This step is usually its own huge section, with different algorithms and methods that can be implemented (i.e., RRT, A*, etc...). It is important that the planner (reference trajectory) outputs an accurate and feasible reference as the MPC controller will try to follow the trajectory as close as possible. For example, an unfeasible reference path is redundant as the controller will not be able to yield a (feasible) solution. To that end, the state and action references were handled differently which will be described below.

For the reference states, the velocity was the only state used. This was the only state used as it keeps the vehicle moving forward and at an appropriate speed (later we will discuss how to keep the vehicle close to the line). Therefore, the reference velocity was set to the desired speed (80 kph) and adjusted according to the upcoming road curvature. When there was a curve in the road greater than a certain threshold, the reference velocity decreased. Additionally, there was a greater velocity decrease if there was a sharp turn.

The actions consisted of driving force and steering angle. To reduce the amount of driving input, we set the desired input to constant zero. It can be noted though that this is in direct conflict with the above desired speed reference state. By putting the input driving force reference to zero, we are preventing the vehicle from saturating its input and not abruptly changing either. Next, the reference steering angle was calculated by reducing the deviation from the center line with a PD-controller. When a deviation from the center line was present, the proportional gain (k_p) would create a steering action to turn the wheel back towards the center line. To prevent drastic turns, a derivative term (k_d) was added which dampened the effects of the input steering reference.

Overall, the above reference provides sufficient (and feasible) information for the vehicle to follow the trajectory.

2.4 Notes for Implementation

Table 1 shows the variable names that were used within the code. Additionally, Table 2 shows the hyperparameters that were chosen for the controller along with the reference controller. Furthermore, please refer to the submitted code to see entire structure and (successful) implementation.

State Variable	Code Observation Name
p	`road_progress_at_closest_point`
d	`distance_to_closest_point`
μ	`heading_angle_relative_to_line`
v	`vx_sensor`

Table 1: State Variable Code Variable Names

Hyperparameter	Value
Q	$\texttt{diag}(0, 0, 0, 10)$
R	$\texttt{diag}(0.1, 0.8)$
N	10
T_s	0.05
k_p	0.02
k_d	0.007

Table 2: MPC Hyperparameters Used

3 Policy Performance

The MPC policy was tested across various tracks and road conditions. Three tracks were designed: one easy, one medium, and one difficult, to assess the models abilities. Additionally, the models were tested in different conditions by modifying parameters to simulate low traction environments, such as wet or icy surfaces. These metrics helped assess the models' adaptability to diverse conditions.

Significant testing was done with the MPC controller. This was done to fine tune the hyperparameters and pushing its capabilities to see what was possible with the controller. Various figures below show some of the tests that were done on increasingly difficult roads. For each of the tests, a desired speed of 80 kph was set. Before diving into the specifics, the controller allows for the vehicle to slow down when there is a turn approaching, and speed up as the turn straightens. Overall, the MPC controller tracked the path very well.

Easy Road

Refer to Figures 1 - 3 for the results of the easy road. This road includes a straight with a wide turn, followed by another straight section, as shown in Figure 1, along with the actual trajectory. If a human was driving, we would expect a slight slow down around the turn, however, not much since it is a wide turn. This behavior is also shown from the controller in Figure 2 as the vehicle slightly reduces its speed by 1.5 kph. It can be seen that the speed does not recover to 80 kph within the time window. This can be reasoned as the drive command reference is set to zero (as mentioned in Section 3.2.3), and the difference in reference velocity and actual velocity is quite small, so the drive input reference has a larger effect (relative to the difference in velocity), and this restricts the input. This pattern can be seen in subsequent runs. Lastly, dive and steering inputs are shown in 3. It should be noted that it looks like sharp deviations, however, the horizontal axis (time) is over a much larger scale which makes the deviations look abrupt. Refer to Table 3 for the performance metrics for varying road conditions.

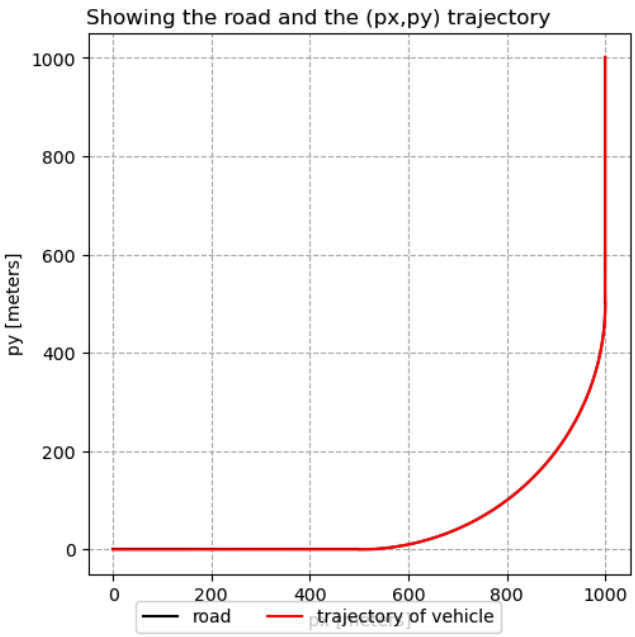

Figure 1: MPC Trajectory - Easy

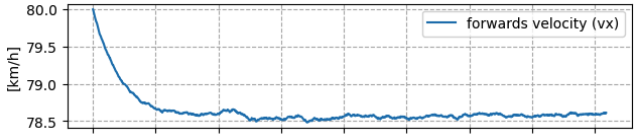

Figure 2: Vehicle Velocity

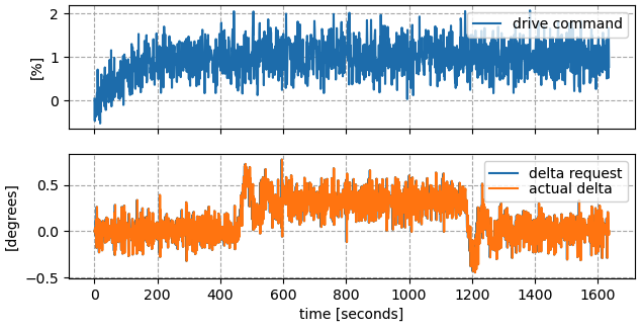

Figure 3: Drive and Steering Inputs

Performance Metric	**Dry**	**Wet**	**Icy**
Drive Length (m)	1784	1784	1784
Average Speed (kph)	78.66	77.59	75.57
Average Deviation (m)	0.13	0.13	0.13
Adaptability	2.07		

Table 3: MPC Performance Metric Varying Conditions - Easy

Medium Road

Refer to Figures 4 - 6. In this track, there are more and sharper turns. As a result, it requires more input than the previous track. The results can be seen in Figure 4, which show a close following of the road. The

variation in the velocity can be see in Figure 5. As there are sharper turns in the road the vehicle slows down substantially more. Similar to above, Figures 5 and 6 have a horizontal axis that spans over a longer duration, which makes the changes look a lot more abrupt. Refer to Table 4 for the performance metrics for varying road conditions.

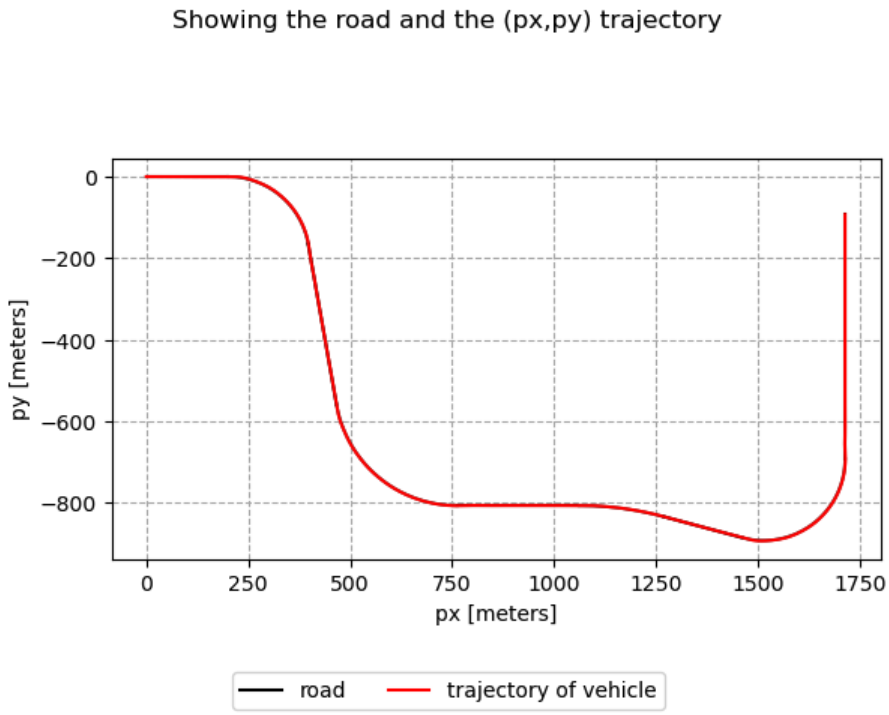

Figure 4: MPC Trajectory - Medium

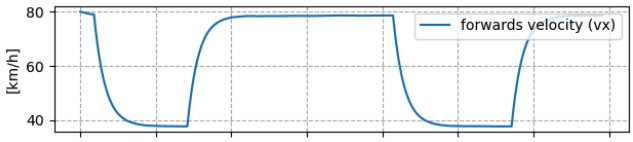

Figure 5: Vehicle Velocity

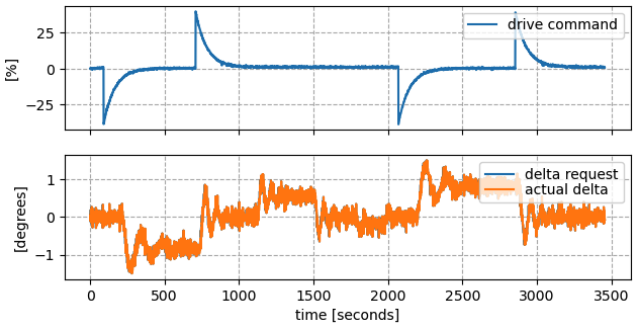

Figure 6: Drive and Steering Inputs

Performance Metric	Dry	Wet	Icy
Drive Length (m)	2973	2973	2973
Average Speed (kph)	64.00	63.69	62.08
Average Deviation (m)	0.34	0.33	0.30
Adaptability	1.11		

Table 4: MPC Performance Metric Varying Conditions - Medium

Difficult Road

Refer to Figures 7 - 9. This road is substantially more challenging than the previous two demonstrated. In this road, there are sharp turns, winding portions, and straight components. Despite its challenges, the controller performed very well as shown in Figure 7. In Figure 8, we see that the vehicle spends a lot more time at 40 kph since there are a sharp turns and more bends, keeping the vehicle safely on the road. Additionally, the drive and steering inputs can be see in Figure 9. Refer to Table 5 for the performance metrics for varying road conditions.

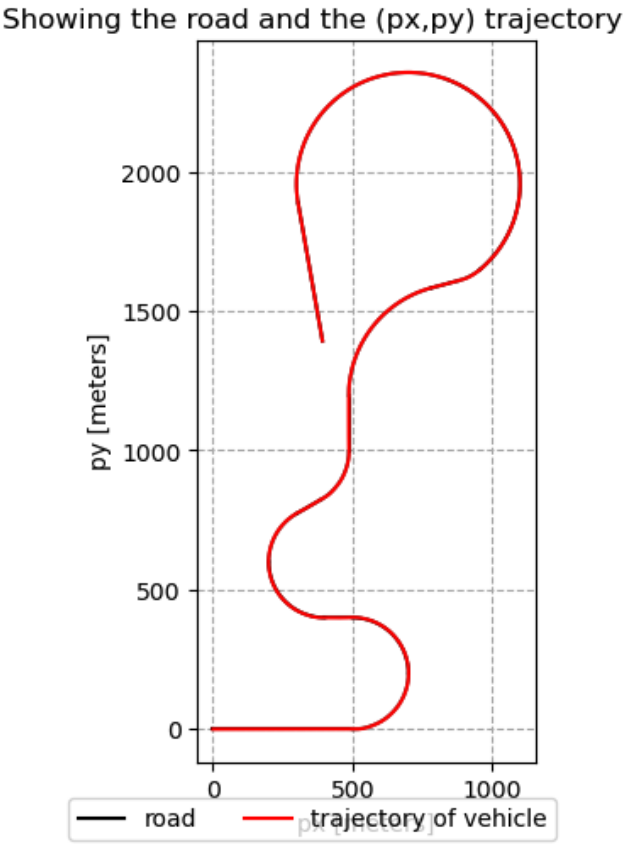

Figure 7: MPC Trajectory - Difficult

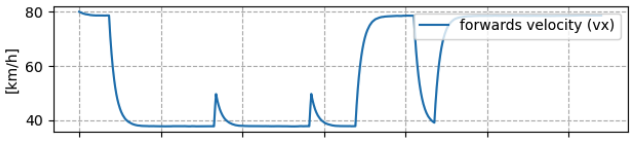

Figure 8: Vehicle Velocity

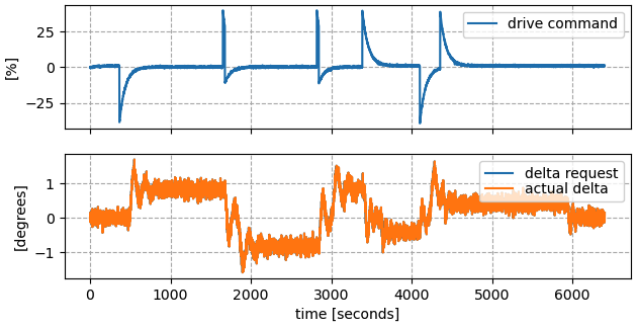

Figure 9: Drive and Steering Inputs

Performance Metric	Dry	Wet	Icy
Drive Length (m)	5147	5147	5147
Average Speed (kph)	57.98	57.88	57.17
Average Deviation (m)	0.44	0.44	0.43
Adaptability	0.45		

Table 5: MPC Performance Metric Varying Conditions - Difficult

Ridiculous Road

A final test was conducted on a road that should have broken the simulation. Refer to Figures 10 and 11. Around the bends, it keeps a slow speed to remain on the track. The speed around the bend is a hyperparameter that can be set within the reference trajectory, which sets a desired speed based on the curvature.

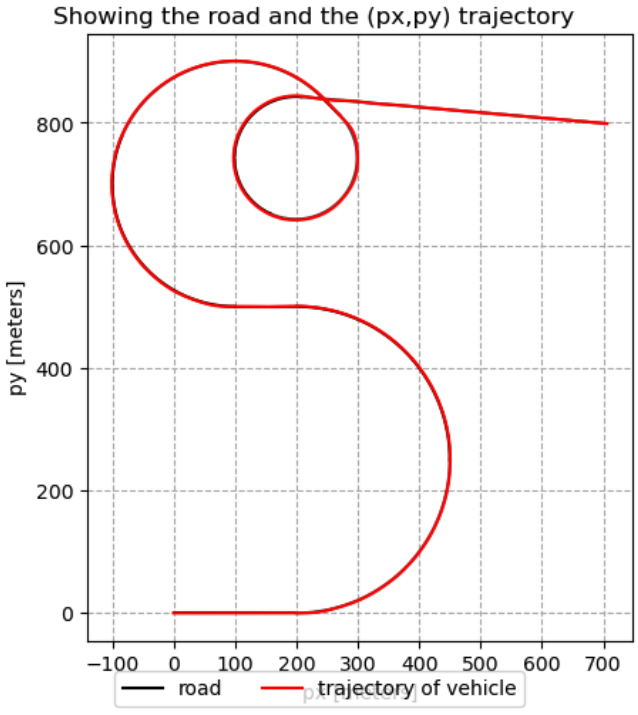

Figure 10: MPC Trajectory - Ridiculous

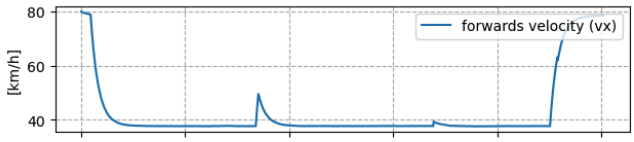

Figure 11: Vehicle Velocity

References

[1] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. D. and, "From linear to nonlinear mpc: Bridging the gap via the real-time iteration," *International Journal of Control*, vol. 93, no. 1, pp. 62–80, 2020. DOI: 10.1080/00207179.2016.1222553.