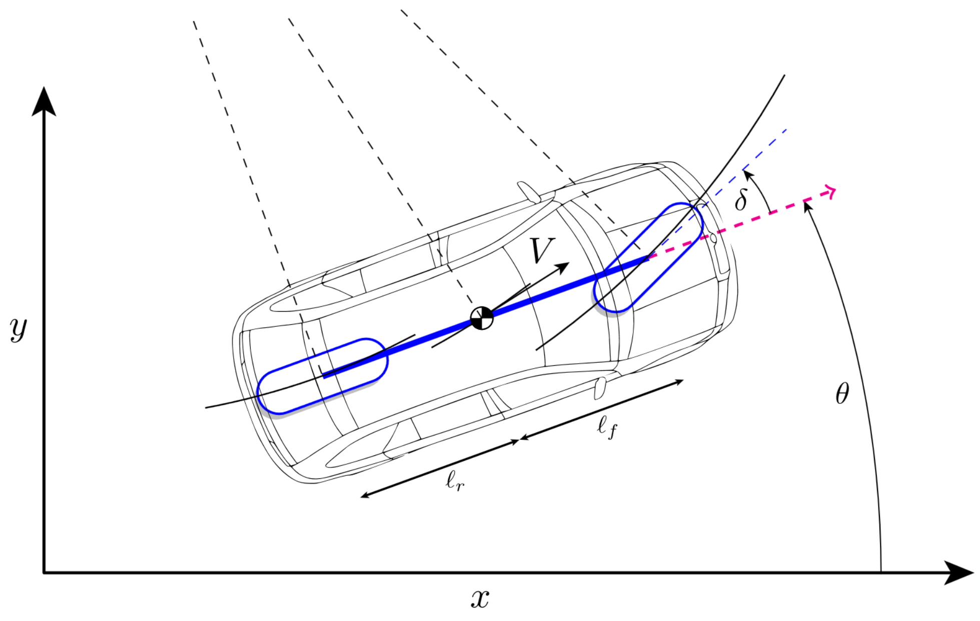
École Polytechnique Fédérale de Lausanne

ME-425 Model Predictive Control

Mini-Project - Group N



Adrien CADET, SCIPER: 346445
Teo HALEVI, SCIPER: 329561
Nevò MIRZAI HAMADANI, SCIPER: 328344

Contents

1 Introduction

In this project, we study how to model, linearize, and control a car using various Model Predictive Control (MPC) strategies. Our main goal is to design controllers that track reference trajectories, respect physical constraints, and handle uncertainties in the vehicle's dynamics. We test these controllers in common highway driving situations, as well as advanced maneuvers such as overtaking.

First, we derive and linearize a state-space model, then separate it into distinct longitudinal and lateral subsystems. We then add an offset-free tracking by adding an observer, followed by a robust tube MPC approach for adaptive cruise control in the presence of disturbances. Finally, we move to a full nonlinear MPC that captures the car's full range of dynamics, including collision avoidance during overtaking. Throughout the project, we describe how we choose parameters, tune the controllers, and measure performance using simulation results.

2 Linearization

2.1 Derivation of analytical expressions of $f(x_s, u_s)$, A, B

The state vector and the input vector are respectively defined as:

$$\boldsymbol{x} = \begin{bmatrix} x & y & \theta & V \end{bmatrix}^T \tag{2.1}$$

$$\boldsymbol{u} = \begin{bmatrix} \delta & u_T \end{bmatrix}^T . \tag{2.2}$$

where x, y represents the position of the car's center of mass in the world frame, θ is the heading angle of the car with respect to the x-axis, and V is the velocity of the vehicle. In the input vector u, δ is the steering angle and u_T is the throttle value. We have

$$A = \left.\frac{\partial f(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x_s}, \boldsymbol{u_s})} \tag{2.3}$$

$$B = \left.\frac{\partial f(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{(\boldsymbol{x_s}, \boldsymbol{u_s})} \tag{2.4}$$

$$\boldsymbol{x_s} = \begin{bmatrix} 0 & 0 & 0 & V_s \end{bmatrix}^T \tag{2.5}$$

$$\boldsymbol{u_s} = \begin{bmatrix} 0 & u_{\mathrm{T,s}} \end{bmatrix}^T \tag{2.6}$$

where the function f is defined as:

$$f(\boldsymbol{x}, \boldsymbol{u}) = \dot{\boldsymbol{x}} = \begin{bmatrix} V\cos(\theta + \beta) \\ V\sin(\theta + \beta) \\ \frac{V}{l_r}\sin(\beta) \\ \frac{F_{motor} - F_{drag} - F_{roll}}{m} \end{bmatrix} \tag{2.7}$$

With this information, we can calculate the partial derivatives of the function f with respect to the state vector.

$$\frac{\partial f(\boldsymbol{x},\boldsymbol{u})}{\partial x} = \boldsymbol{0} \tag{2.8}$$

$$\frac{\partial f(\boldsymbol{x},\boldsymbol{u})}{\partial y} = \boldsymbol{0} \tag{2.9}$$

$$\frac{\partial f(\boldsymbol{x},\boldsymbol{u})}{\partial \theta} = \begin{bmatrix} -V\sin(\theta+\beta) \\ V\cos(\theta+\beta) \\ 0 \\ 0 \end{bmatrix} \tag{2.10}$$

$$\frac{\partial f(\boldsymbol{x},\boldsymbol{u})}{\partial V} = \begin{bmatrix} -\cos(\theta+\beta) \\ \sin(\theta+\beta) \\ \dfrac{\sin(\beta)}{l_r} \\ -\dfrac{u_T P_{max}}{mV^2} - \dfrac{\rho C_d A_f V}{m} \end{bmatrix} \tag{2.11}$$

We notice that $\beta_s = 0$ because $\delta_s = 0$, and therefore,

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & V_s & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\dfrac{u_{T,s}P_{max}}{mV_s^2} - \dfrac{\rho C_d A_f V_s}{m} \end{bmatrix} \tag{2.12}$$

Moreover, we can compute

$$\frac{\partial \beta}{\partial \delta} = \frac{l_r}{l_r + l_f} \cdot \frac{1}{\cos^2\delta} \cdot \frac{1}{1 + \left(\dfrac{l_r \tan\delta}{l_r + l_f}\right)^2} \tag{2.13}$$

That implies that:

$$\left.\frac{\partial \beta}{\partial \delta}\right|_{\delta=0} = \frac{l_r}{l_r + l_f} \tag{2.14}$$

We obtain the following results, applying the same reasoning to compute the partial derivatives along $\boldsymbol{u}$:

$$B = \begin{bmatrix} 0 & 0 \\ \dfrac{l_r}{l_r + l_f} V_s & 0 \\ \dfrac{1}{l_r + l_s} V_s & 0 \\ 0 & \dfrac{P_{max}}{V_s m} \end{bmatrix} \tag{2.15}$$

The linearized dynamics of the system are found by a Taylor series expansion:

$$f(\boldsymbol{x},\boldsymbol{u}) \approx f(\boldsymbol{x_s},\boldsymbol{u_s}) + A(\boldsymbol{x} - \boldsymbol{x_s}) + B(\boldsymbol{u} - \boldsymbol{u_s}) \tag{2.16}$$

The final analytical expression of $f(\boldsymbol{x_s},\boldsymbol{u_s})$ is then:

$$f(\boldsymbol{x_s},\boldsymbol{u_s}) = \begin{bmatrix} V_s \\ 0 \\ 0 \\ \dfrac{u_{T,s}P_{max}}{mV_s} - \dfrac{\rho C_d A_f V_s^2}{2m} - C_r g \end{bmatrix} \tag{2.17}$$

2.2 Separation into 2 independent subsystems

The system is divided into two parts: one for longitudinal motion (x and V) and another for lateral motion (y and θ). This separation is based on the different physical and mechanical factors influencing each.

Longitudinal Subsystem
This part deals with forward motion, including the position (x) and velocity (V). It is controlled by the throttle input (u_T), which affects how fast the car moves forward. The forward motion depends on forces like engine power, air resistance, and rolling friction. These forces do not interact with the car's steering or lateral motion, making this subsystem independent.

Lateral Subsystem
This part focuses on sideways motion and direction, described by the lateral position (y) and orientation (θ). It is controlled by the steering angle (δ), which determines the car's turning path. These variables depend only on the steering geometry and are not influenced by the forces driving the car forward, keeping this subsystem separate.

By analyzing matrices A and B derived above, it becomes clear that the longitudinal variables (x, V) are determined only by the state V and the throttle u_T. Similarly, one can see that the lateral variables (y, θ) are influenced by themselves and the steering δ.

The independence of these two subsystems comes from the way forces act on the car. Forward motion is controlled by forces along the car's main axis (like thrust and drag), while lateral motion depends on steering, which creates turning and sideways movement. This separation makes it easier to analyze and control each subsystem individually.

3 MPC Controllers for Each Sub-System

3.1 Design procedure to ensure recursive constraint satisfaction

Recursive constraint satisfaction guarantees that system constraints are met at every step in the prediction horizon. After linearization, the discrete-time dynamics are given by:

$$\boldsymbol{x}^+ = f_d(\boldsymbol{x_s}, \boldsymbol{u_s}) + A_d(\boldsymbol{x} - \boldsymbol{x_s}) + B_d(\boldsymbol{u} - \boldsymbol{u_s}) \tag{3.18}$$

and allow deriving the prediction model for longitudinal and lateral controllers. We linearize the system around $V = 120$ km/h as in all the project parts.

3.1.1 Optimization problem

The optimization problem for longitudinal and lateral controllers is formulated to minimize a weighted sum of state and input deviations:

$$\min_{x,u} \sum_{i=0}^{N-1} \left[(x_i - x_{\text{ref}})^T Q (x_i - x_{\text{ref}}) + (u_i - u_{\text{ref}})^T R (u_i - u_{\text{ref}})\right] + J_{\text{terminal}}$$

Subject to:

$$x_{i+1} = f_{xs\_us} + A(x_i - x_s) + B(u_i - u_s)$$

$$Mu_i \leq m$$
$$Fx_i \leq f$$

$$F_f(x_N - x_{\text{ref}}) \leq f_f$$

3.1.2 Longitudinal constraints

The input constraint is:

$$-1 \leq u_T \leq 1 \tag{3.19}$$

Since the two states for this subsystem are x and V, we do not have any state constraint.

Expressing input constraints in the form $Mu \leq m$, we define:

$$M = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad m = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{3.20}$$

3.1.3 Lateral constraints

The input constraint on the second subsystem concerns the steering wheel angle δ:

$$|\delta| \leq 30^\circ = 0.5236 \text{ rad} \tag{3.21}$$

Using the previous notation, we express it as:

$$M = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad m = \begin{bmatrix} 0.5236 \\ 0.5236 \end{bmatrix} \tag{3.22}$$

For lateral control, we have two constraints on the state variables y and θ:

$$-0.5\text{m} \leq y \leq 3.5\text{m} \tag{3.23}$$

$$|\theta| \leq 5^\circ = 0.0873 \text{ rad} \tag{3.24}$$

These state constraints are expressed in matrix form:

$$F = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix}, \quad f = \begin{bmatrix} 3.5 \\ 0.5 \\ 0.0873 \\ 0.0873 \end{bmatrix} \tag{3.25}$$

3.1.4 Terminal set and terminal cost

The Linear Quadratic Regulator (LQR) provides the foundation for defining the terminal cost and the terminal controller in our problem. The goal of the LQR is to stabilize the system around an equilibrium point by minimizing a quadratic cost function J that penalizes state deviations and control effort. The cost function is given by:

$$J = \sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k \tag{3.26}$$

subject to

$$x_{k+1} = Ax_k + Bu_k$$

The solution involves solving the Discrete Algebraic Riccati Equation (DARE) to obtain the optimal feedback gain K_f and the associated cost matrix P. Stability is achieved by applying the terminal control law $u = K_f x$, which gives the closed-loop system dynamics $x_{k+1} = (A + BK_f)x_k$. This stability is used to define the terminal cost as $V_f(x_N) = x_N^T P x_N$, representing the long-term effect of state deviations. We can thus define the terminal cost for both controllers as:

$$J_{\text{terminal}} = (x_N - x_{\text{ref}})^T P (x_N - x_{\text{ref}}) \tag{3.27}$$

The next step is to define the terminal set $\mathcal{X}_f$. This set ensures that the terminal state is in a region where the LQR controller can keep the system within constraints forever, making sure the solution remains valid at every step. $\mathcal{X}_f$ is constructed for the lateral controller only to satisfy control invariance and constraints satisfaction. Respectively:

$$x_k \in \mathcal{X}_f \implies x_{k+1} = (A + BK_f)x_k \in \mathcal{X}_f$$
$$Fx_k \leq f, \quad MK_f x_k \leq m, \quad \forall x_k \in \mathcal{X}_f$$

$\mathcal{X}_f$ is computed iteratively using the algorithm seen in the course (slide 4-29). We can thus state that $\mathcal{X}_f = \{x|F_f x \leq f_f\}$, where F_f and f_f define the terminal polyhedron.

3.1.5 Recursive feasibility

This design of optimization problem ensures the closed-loop system stability because:

- The stage cost is a positive definite function.
- The terminal set is invariant under the terminal control law and all state and input constraints are satisfied in $\mathcal{X}_f$. And the input and state constraints are satisfied throughout the horizon.
- The terminal cost is a continuous Lyapunov function in the terminal set $\mathcal{X}_f$.

If the optimization problem is feasible at time step k, it remains feasible at time step $k+1$, ensuring continuity in control actions.

3.2 Choice of tuning parameters (Q, R, H)

In this section, we discuss the tuning of the parameters Q and R for the `MpcControl_lon` and `MpcControl_lat` files, as well as the horizon H we use in the simulation.

3.2.1 Parameters of the longitudinal controller

The matrix Q penalizes deviations of the system state from the reference value and is defined as

$$Q = 25$$

This value is one-dimensional because the state involves only one key component (velocity V). The reason is that there are no cost or constraints on the x-position. Hence, there is no need to optimize over the x-position (which is also not a steady-state) in the longitudinal case.

The matrix R penalizes the magnitude of the control input to ensure smooth operation and is defined as

$$R = 0.5$$

The choice of these two values results in a speed response that rapidly converges towards the desired speed of 120km/h (cf. Fig. 3.2, Velocity plot).

3.2.2 Parameters of the lateral controller

Lateral controller parameters are simpler. We define :

$$Q = \begin{bmatrix} 5 & 0 \\ 0 & 20 \end{bmatrix}$$

$$R = 25$$

These values ensure passenger comfort by limiting sudden changes in steering wheel direction. The results obtained are unchanged in terms of performance, but give a smoother result compared to the identity values of Q and R.

3.2.3 Prediction horizon H

The prediction horizon H satisfy the equation

$$H = N \cdot T_s$$

where N is the number of steps in the horizon, and T_S is the sampling time. H is proportional to the horizon N and thus has influence on the speed of convergence and the region of attraction. We determine by simulation that the limit value of H to ensure a solution of the problem by the optimizer is $H = 0.9$s. If H is too small the problem becomes unfeasible, indeed the optimizer can not find any solution to reach the terminal set in N steps. Conversely, a too high H results in increasing the computational cost. The value of H does not significantly change the computation time of our simulation, and we therefore decide to define $H = 10$ seconds. With $T_s = 1/10$ seconds, we obtain a "true" horizon of $N = 100$ steps which seems to be a good trade-off between computational time and accuracy.

3.3 Terminal invariant set for lateral sub-system

The terminal invariant set for the lateral subsystem was obtained by computing the maximum invariant set of the lateral subsystem under the control law $u_{\text{lat}} = K_f x_{\text{lat}}$, where K_f is the optimal LQR controller for the matrices Q and R defined above. This terminal set was computed using the algorithm seen in the course (see slide 4-29).

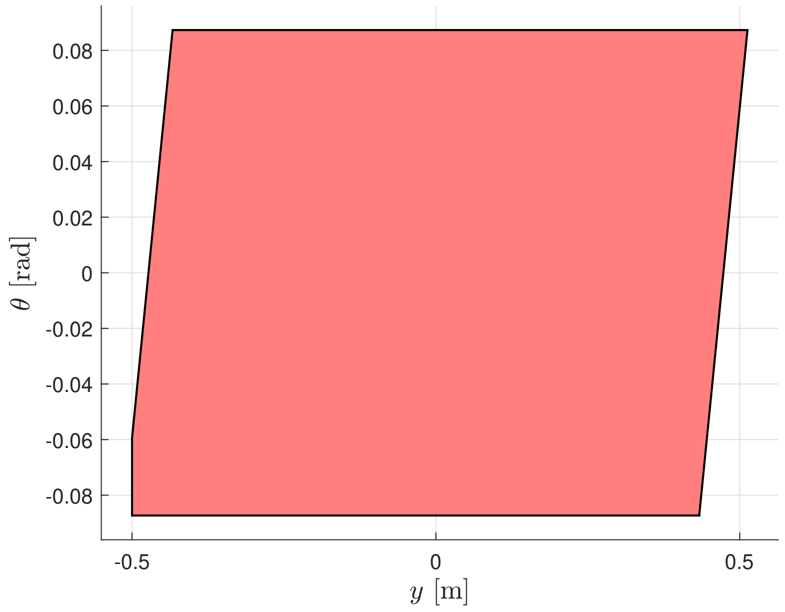


Fig. 3.1 | Terminal invariant set for the lateral sub-system

3.4 Closed-loop plots for each dimension

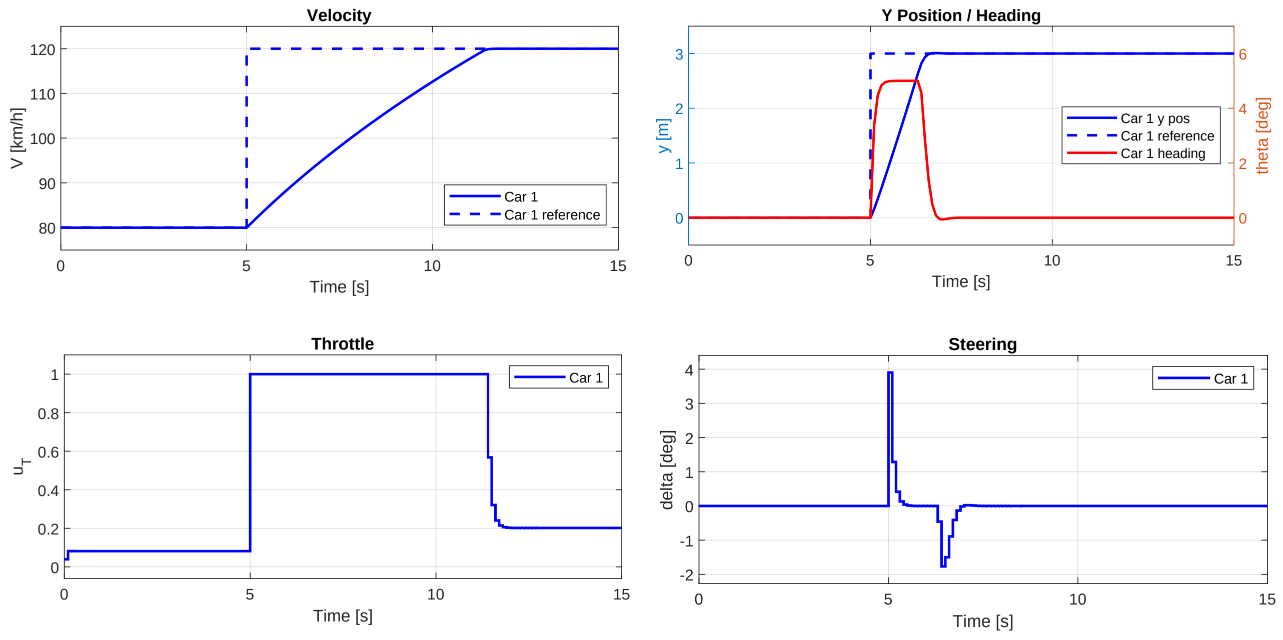


Fig. 3.2 | Simulation results showing the vehicle's longitudinal and lateral dynamics.

The figure 3.2 presents the simulation results for the vehicle's longitudinal and lateral dynamics. The top-left plot shows the velocity V, which follows the reference trajectory (dashed line) with a transient phase caused by system dynamics. The top-right plot illustrates the lateral position y and heading angle θ. y closely tracks the reference, while θ stabilizes after an initial correction. The bottom-left plot shows the throttle input u_T, which increases rapidly to reach the desired velocity, then decreases gradually to maintain steady-state operation. The bottom-right plot presents the steering angle δ which adjusts dynamically to track the lateral reference.

These graphs clearly show that the constraints on settling time have been respected. The 120km/h speed is reached in less than 10 seconds, and the lane change takes less than 3 seconds. More precisely, the reference speed is reached in just under 7 seconds, and the lane change is completed in just over 1.5 seconds.

4 Offset-Free Tracking

4.1 Design procedure and choice of tuning parameters

The goal of this design is to create an observer that estimates both the vehicle's velocity (V) and an unknown constant disturbance (d) affecting the system. This ensures offset-free tracking in the presence of model uncertainties or external disturbances.

Recall that the linearization error enters the dynamics of the system via the input u_T according to

$$\boldsymbol{x}^+ = f_d(\boldsymbol{x_s}, \boldsymbol{u_s}) + A_d(\boldsymbol{x} - \boldsymbol{x_s}) + B_d(\boldsymbol{u} - \boldsymbol{u_s}) + \hat{B}_d d \tag{4.28}$$

where $\hat{B}_d$ is the submatrix of B_d for the input u_T.

4.1.1 Extended State-Space Model

The first step is to augment the system dynamics to include the disturbance d as an additional state. The extended state vector is:

$$\hat{z} = \begin{bmatrix} V \\ d \end{bmatrix}$$

The dynamics of the extended system are represented as:

$$\begin{aligned} \hat{z}_{k+1} &= \hat{x}_s + \hat{A}(\hat{z}_k - \hat{x}_s) + \hat{B}(u_k - \hat{u}_s) && (4.29) \\ &= \hat{x}_s + \begin{bmatrix} A_{d,22} & B_{d,2} \\ 0 & 1 \end{bmatrix} (\hat{z}_k - \hat{x}_s) + \begin{bmatrix} B_{d,2} \\ 0 \end{bmatrix} (u_k - \hat{u}_s) && (4.30) \end{aligned}$$

The disturbance d is assumed to evolve as a constant, leading to the inclusion of 1 in the lower-right corner of $\hat{A}$. The lower component of $\hat{B}$ is zero because input u only affects the velocity V.

4.1.2 Observer Design

The observer is designed to estimate $\hat{z} = [V, d]^T$ based on the measurements $y = V$. The observer dynamics including the steady-state offsets are given by:

$$\hat{z}_{k+1} = \hat{x}_s + \hat{A}\left(\hat{z}_k - \hat{x}_s\right) + \hat{B}\left(u_k - \hat{u}_s\right) + L\left(\hat{C}\hat{z}_k - y\right) \tag{4.31}$$

The gain matrix L is calculated using the pole placement method with the MATLAB function `place`:

$$\text{eig}(\hat{A} + L\hat{C}) = \{p_1, p_2\} = \{0.75, 0.9\} \tag{4.32}$$

The poles $p_1 = 0.75$ and $p_2 = 0.9$ are chosen to achieve a balance between convergence speed of the estimated states (poles closer to zero ensure faster convergence) and robustness (poles not too close to zero avoid high sensitivity to noise or model uncertainties).

4.1.3 Parameters Q and R

The weighting matrices for the cost function are defined as follows. Matrix Q of the lateral controller is identical to part 3. The value of R has been increased to compensate for the impact of other parameters on comfort when changing direction. We therefore define:

$$Q = \begin{bmatrix} 5 & 0 \\ 0 & 20 \end{bmatrix}$$

$$R = 50$$

The challenge of Q and R values for the longitudinal controller is to be able to track the reference speed without offset, while limiting oscillations and overshoot on the throttle and speed. The values we have found are:

$$Q = 2, \quad R = 2$$

These values give a speed offset of 0.006 km/h on the reference speed, which we consider as offset-free.

To track speed even better, we can increase the value of Q (up to $Q = 5$ according to our simulations). The speed reached is then identical to the reference value with a margin of error of 0.004km/h, but induces slightly greater oscillations and overshoot.

4.2 Impact of the state estimator

We simulate the system for 15 seconds, with a starting state $\boldsymbol{x_0} = \begin{bmatrix} 0 & 0 & 0 & 80/3.6 \end{bmatrix}'$, and a reference to reach $\begin{bmatrix} y & V \end{bmatrix}' = \begin{bmatrix} 3 & 50/3.6 \end{bmatrix}'$. The reference step is delayed by 2 seconds.

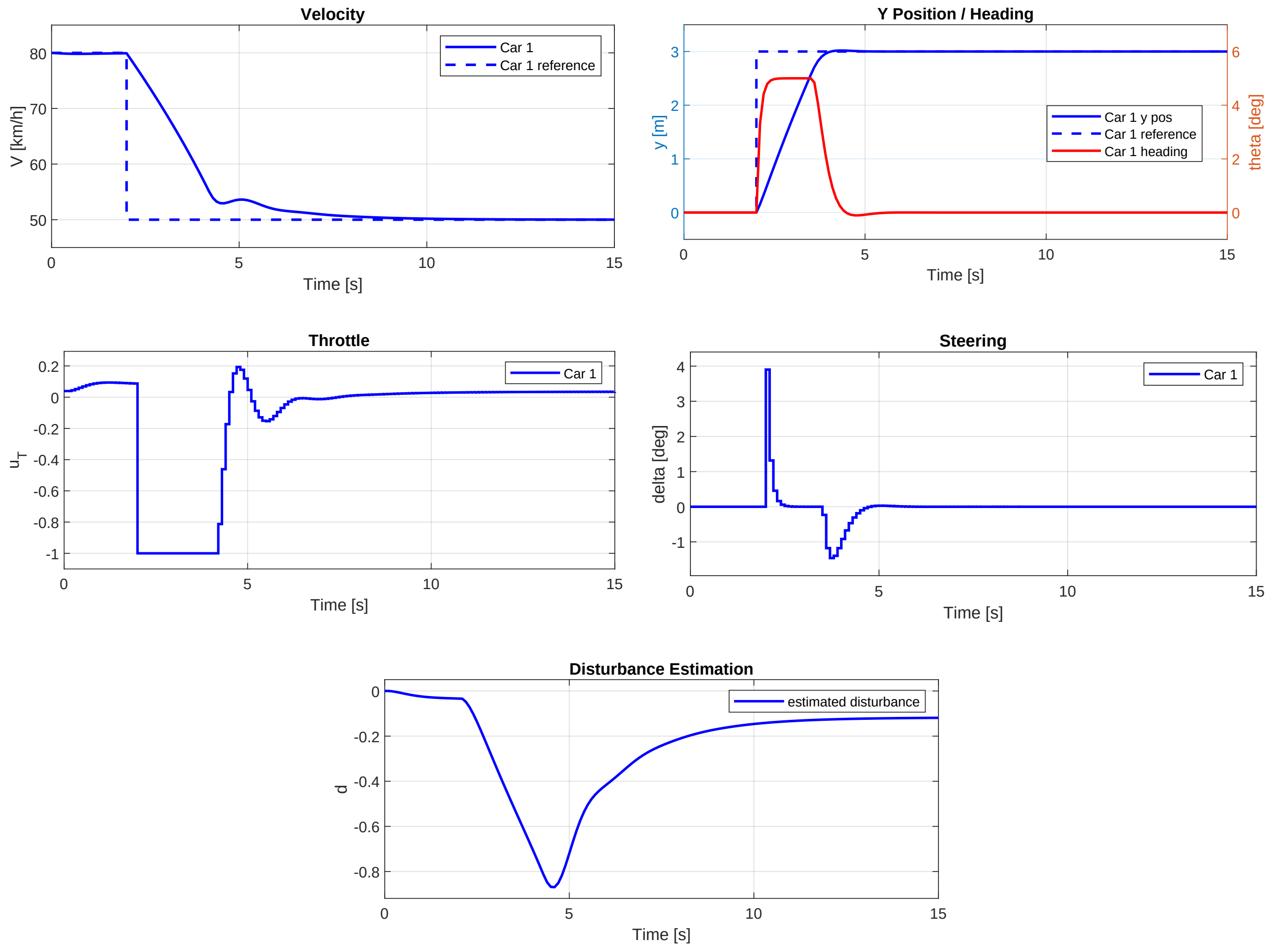


Fig. 4.3 | Simulation results demonstrating the impact of the state estimator

The top-left plot of the figure 4.3 shows the velocity V tracking the reference trajectory (dashed line) with offset-free performance after disturbance rejection. The top-right plot illustrates the lateral position y accurately following its reference and the heading angle θ stabilizing after an initial transient. The middle-left plot shows the throttle input u_T, where adjustments compensate for disturbances to maintain the desired velocity. The middle-right plot presents the steering angle δ, which ensures precise lateral control. The bottom-center plot presents the estimated disturbance d, showing how the state estimator compensates for disturbances during the simulation.

The disturbance estimation plot illustrates the system's ability to identify and compensate for external disturbances affecting the ego car's motion. Initially, a significant drop in the estimated disturbance d causing deviations from the desired trajectory. Between 3 and 5 seconds, the system gradually adapts as the estimator and controller compensate for the disturbance, ensuring that the car follows the reference trajectory accurately. From 5 seconds, the estimated disturbance stabilizes near zero, indicating that the disturbance has been fully rejected, and the system has reached steady-state behavior.

5 Robust Tube MPC for Adaptive Cruise Control

The aim of this part is to create a robust tube MPC controller for adaptive cruise control. The ego car, so as to be able to follow the lead car as closely as possible. The ego car is controlled to regulate the distance and relative speed of the two cars. In the following sections, the tilde refers to the lead car.

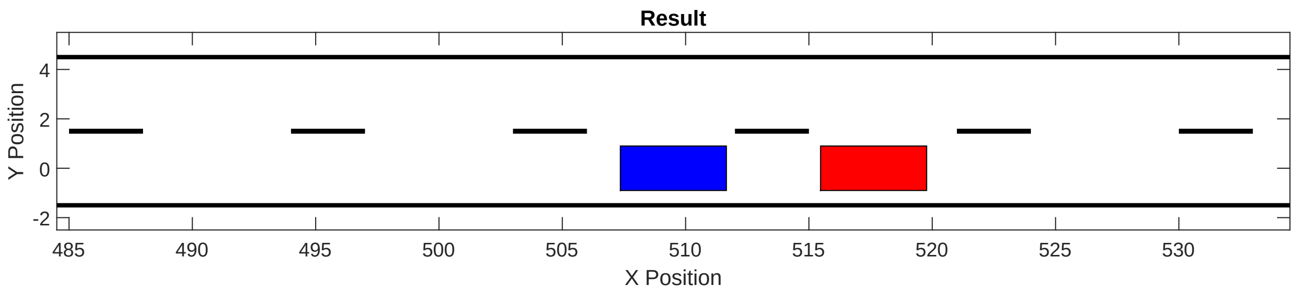


Fig. 5.4 | Illustration of the ego car (blue) following the lead car (red)

5.1 Design procedure and choice of tuning parameters

The first part consists in defining the constraints governing the behavior of the two cars. The ego car has the same constraints as those defined in parts 3.1.2 and 3.1.3. The behavior of the lead car is considered to be a bounded disturbance at the longitudinal level. Its throttle is constrained as follows:

$$\tilde{u}_T \in [u_{T,s} - 0.5, u_{T,s} + 0.5] \tag{5.33}$$

where $u_{T,s}$ is the linearization steady-state throttle input.

The constraint to avoid a collision between the two cars is expressed by:

$$\tilde{x} - x \geq 6 \tag{5.34}$$

The implementation differs from the previous parts as we implemented an MPC controller in a different formulation. We indeed formulated the problem as stated in slide 8-33 ensuring that constraints are robustly satisfied and that the closed-loop system is robustly stable.

5.1.1 Minimum Robust Invariant Set

The computation of the Minimum Robust Invariant Set $\mathcal{E}$ defines the bounds on disturbance-induced deviations in the system state. The disturbance in our problem arises from the lead car's throttle input, defined in equation (5.33). The uncertainty is propagated through the system's dynamics using the input matrix B_d, which connects disturbances to the state evolution.

To represent the uncertainty, a disturbance set $\mathcal{W}$ is defined as:

$$\mathcal{W} = \{w \mid Mw \leq m\},$$

where

$$M = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix}, \quad m = \begin{bmatrix} B_d(1)(u_{T,s} + 0.5) \\ -B_d(1)(u_{T,s} - 0.5) \\ B_d(2)(u_{T,s} + 0.5) \\ -B_d(2)(u_{T,s} - 0.5) \end{bmatrix}.$$

$B_d(1)$ and $B_d(2)$ are the longitudinal components of the input matrix B_d. These coefficients scale the disturbance bounds defined by $u_{T,s} \pm 0.5$ to quantify the impact on each dimension of the system's state.

The tracking controller gain K regulates the system state toward the tube center, mitigating the effects of disturbances. Using pole placement, the controller gain was computed for closed-loop poles at $\{0.9, 0.8\}$ to achieve desired stability properties:

$$K = \texttt{place}(A_d, B_d, [0.9, 0.8]).$$

The resulting closed-loop dynamics are given by:

$$A_{\text{new}} = A_d - B_d K.$$

The Minimum Robust Invariant Set $\mathcal{E}$ is computed iteratively thanks to the algorithm slide 8-24 and we used the slide 8-25 to compute the Minkowsky Sums for Polyhedrons. To manage the complexity of $\mathcal{E}$, its representation was minimized after each iteration using `minHRep`. The iterations were terminated when the norm of the propagated state contributions became negligible:

$$\|A_{\text{new}}^i\| < 10^{-2}.$$

The final set $\mathcal{E}$ represents the smallest invariant set that contains all possible states under the influence of bounded disturbances.

5.1.2 Tightened constraints

The system constraints are tightened by subtracting the invariant set $\mathcal{E}$ from the original constraints. This guarantees that the system's trajectories remain within feasible bounds even under the worst-case disturbance.

The safety margin, x_{safe}, was chosen to enforce a minimum distance of 6 m between the ego and lead cars, as per equation (5.34). Including an additional buffer, x_{safe} was set to:

$$x_{\text{safe}} = \begin{bmatrix} 8 \\ 0 \end{bmatrix}.$$

The tightened state constraints were computed as:

$$\tilde{\mathbb{X}} = \mathbb{X} \ominus \mathcal{E},$$

where

$$\mathbb{X} = \{z \mid Fz \le f\}, \quad F = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix}, \quad f = \begin{bmatrix} 400 \\ x_{\text{safe}}(1) - 6 \\ 700 \\ 700 \end{bmatrix}.$$

The input constraints were similarly tightened by accounting for the feedback controller's impact on $\mathcal{E}$:

$$\tilde{\mathbb{U}} = \mathbb{U} \ominus K\mathcal{E},$$

where

$$\mathbb{U} = \{u \mid Hu \le h\}, \quad H = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad h = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

5.1.3 Terminal Components

The terminal components ensure the closed-loop stability of the MPC controller. Quadratic cost matrices penalize state deviations and control effort:

$$Q = 15I, \quad R = 1.$$

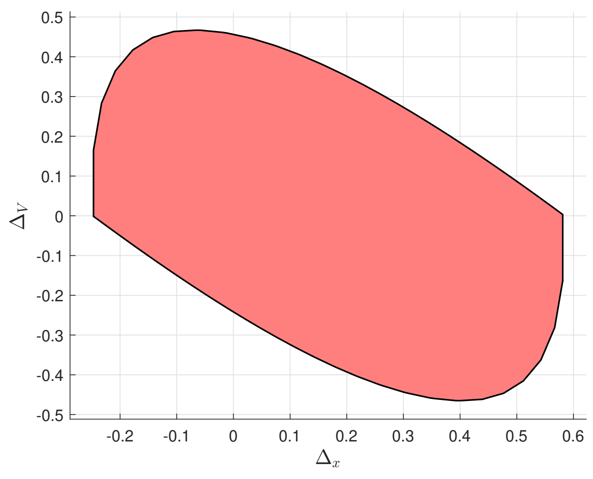
A terminal feedback gain K_f and terminal cost Q_f were computed using the discrete Linear Quadratic Regulator (LQR) approach:

$$[K_f, Q_f] = \texttt{dlqr}(A_d, -B_d, Q, R),$$

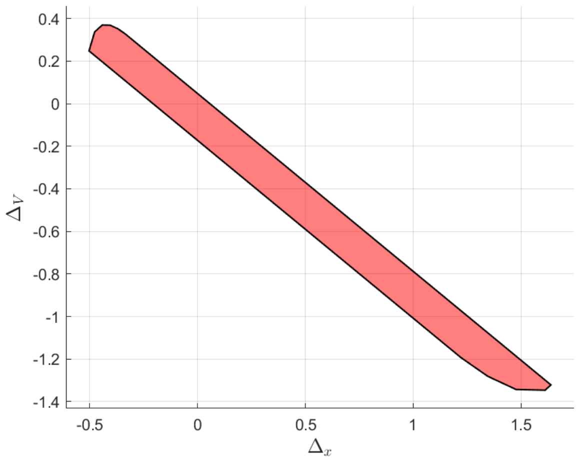
where K_f was applied with an inverted sign because the Matlab command dlqr defines the feedback matrix K to be -K.

The terminal set $\mathcal{X}_f$ was computed iteratively again thanks to the algorithm in the slide 4-29. However, this maximum invariant set was computed with respect to the the tightened constraints under the control law $u = K_f x$. This guarantees that the ego car can be brought to a safe state under the terminal controller, ensuring robust stability.

5.2 Minimal invariant set $\mathcal{E}$ and terminal set $\mathcal{X}_f$



a Minimal invariant set $\mathcal{E}$



b Terminal set $\mathcal{X}_f$

Fig. 5.5

The choice of the tracking controller leading to that minimal invariant set $\mathcal{E}$ was a bit of a trade-off. Indeed, our goal was to find a K that would reduce the size of $\mathcal{E}$, while maintaining a good control authority available to control our tube centers in $\tilde{\mathbb{U}}$. This is because a K that reduces the size of $\mathcal{E}$ is usually a "bigger" K. The trade-off lies in the $K\mathcal{E}$ product, that we would like to make rather small in order to increase the available control authority of our tube centers. The terminal set is invariant for the nominal system and contained in the tightened constraints satisfying one of the three Tube-MPC assumptions.

5.3 Results

First, we can see that with the lead car at a constant throttle, the velocity increases rapidly to get closer to the lead car before stabilizing itself to the value of the lead car. Also, we can note that the distance to the lead car converges to 8 m as we expected because we have set $x_{safe,pos} = 8$. When the lead car has a varying throttle, we can observe first that the constraints are satisfied and, notably, that the distance between the two cars never falls below 6 meters. However, the ego car is experiencing some aggressive changes in throttle, making it track the speed of the car in front. This is not ideal for comfort, but as we want to follow the lead car fairly closely, this discomfort could not be avoided. The resulting figures are shown below.

5.3.1 Robust controller from 100 km/h to 120 km/h following the lead car at constant throttle

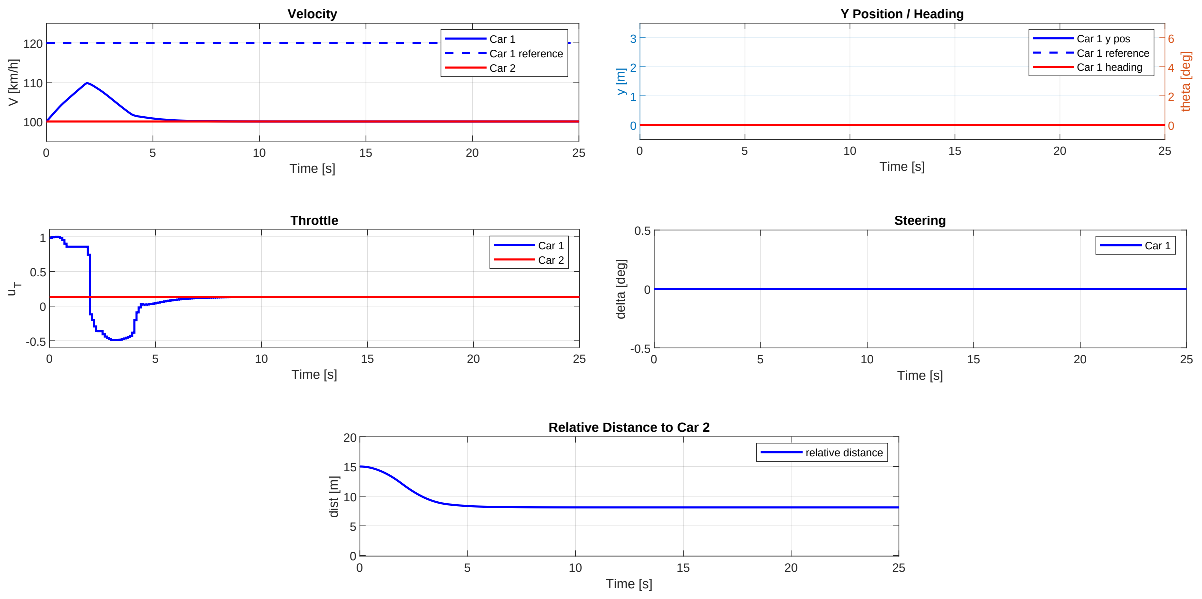


Fig. 5.6 | Constant Lead

5.3.2 Robust controller from 115 km/h to 120 km/h following the lead car at varying throttle

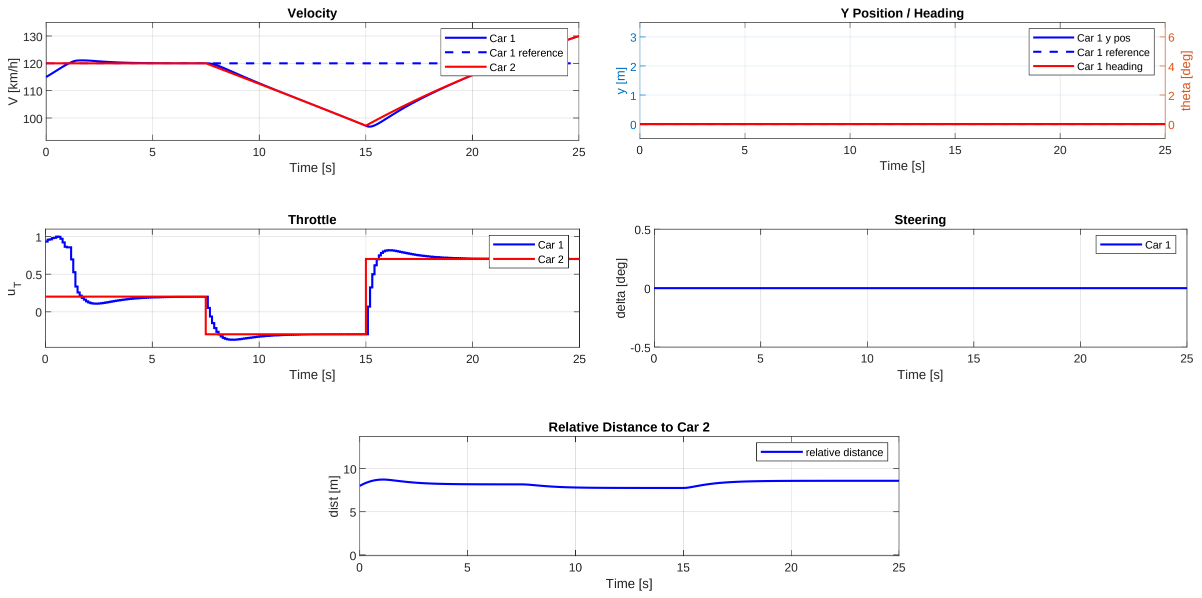


Fig. 5.7 | Varying Lead

6 Nonlinear MPC (NMPC)

6.1 Design Tracking Nonlinear MPC Controller (Deliverable 6.1)

In this section, we developed a nonlinear MPC controller for the ego car to overtake the lead car using CasADi. The NMPC controller operates on the nonlinear model in Equation 2.7 and can therefore cover the whole state space.

6.1.1 Design procedure and choice of tuning parameters

We start with a 4-state car model $\mathbf{x} = [x,\, y,\, \theta,\, V]^\top$ and inputs $\mathbf{u} = [\delta,\, u_T]^\top$. In `NmpcControl.m`, we discretize the model using a 4th order Runge-Kutta step (RK4) of sample period h. We define a horizon $N = \lceil \frac{H}{T_s} \rceil + 1$, where H is the total prediction horizon in seconds, and T_s is the sampling time.

Constraints. We impose the same constraints as in section 3.1.2 and 3.1.3. In addition, each $\mathbf{X}_{k+1}$ must follow the RK4 discretized dynamics given by

$$\mathbf{X}_{k+1} = f_{\text{discrete}}\big(\mathbf{X}_k,\, \mathbf{U}_k\big).$$

This makes the trajectory respect the nonlinear model.

Settling Time Requirement. We want the system to:

1. Accelerate from 80 to 100 km/h in ≤ 5 s,
2. Perform a lane change from $y = 0$ to $y = 3$ m also in less than 5s.

We found that a horizon $H = 10$ seconds was suitable for the task.

Cost Function. We changed the matrices Q and R to:

$$Q = \text{diag}(0,\, 1,\, 1,\, 1), \quad R = \text{diag}(2000,\, 1)$$

As in linear MPC, there is no cost or constraint on the x-position. Therefore, the first value of Q is set to 0. Initially, all other values in Q and R were set to 1. After testing the code, we observed that the steering behavior was overly aggressive; the acceleration experienced in the car seemed excessively strong. For this reason, the R value associated with steering was increased to 2000, reducing the sharpness of the peak in the steering angle and thus making the lane change smoother.

6.1.2 Steady-state tracking error

We know that the easiest way to remove any remaining steady-state error would be to increase the last value in the Q matrix, the one linked to following the desired velocity. We noticed, though, that there was always (even if minimal) some steady-state error remaining, so, mainly for the challenge, we tried to remove it completely. Initially, we formulated our input cost term in the cost function as follows:

$$(\mathbf{U}_k)^\top R\, (\mathbf{U}_k),$$

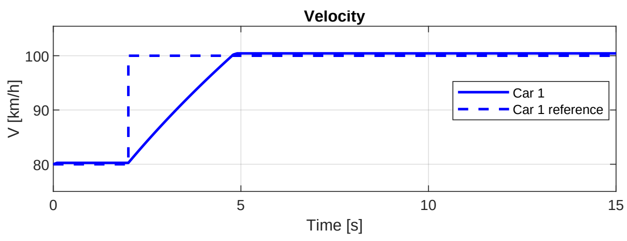
In order to overcome the remaining steady-state error, we thought of formulating the problem differently: we therefore introduced `Ut_ref` (the input needed to keep the desired velocity) to help remove it. We penalize

$$\big(\mathbf{U}_k - [0,\, \texttt{Ut\_ref}]^\top\big)^\top R\, \big(\mathbf{U}_k - [0,\, \texttt{Ut\_ref}]^\top\big),$$

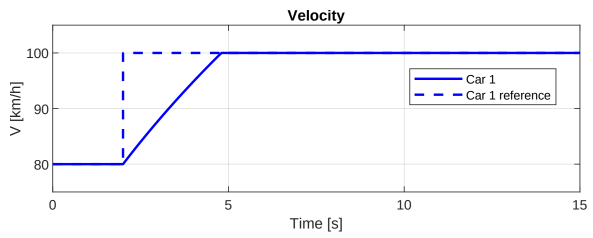
which forces the throttle u_T to settle near the offset value required to keep $V \approx V_{\text{ref}}$. Thus, each predicted step k in the cost is:

$$(\mathbf{X}_k - \mathbf{x}_{\text{ref}})^\top Q\,(\mathbf{X}_k - \mathbf{x}_{\text{ref}}) \;+\; (\mathbf{U}_k - [\,0,\ \texttt{Ut\_ref}\,])^\top R\,(\mathbf{U}_k - [\,0,\ \texttt{Ut\_ref}\,]).$$

Because the controller penalizes $(y - y_{\text{ref}})$ and $(V - V_{\text{ref}})$ in the cost, the system converges close to the reference under nominal conditions. However, the main reason we see a bigger offset without `Ut_ref` is that the controller, despite penalizing $(V - V\text{ref})$, lacks an explicit mechanism (like integral action or an offset input) to hold the exact throttle required for steady-state at V_{ref}. Therefore, minimizing $(V - V_{\text{ref}})$ alone does not force V to end up exactly at V_{ref} if the system dynamics require a small steady input offset to maintain that speed (especially under unmodeled disturbances or drag forces). By adding `Ut_ref`, we supply that missing offset term in the optimization, completely eliminating the steady-state speed error. By including `Ut_ref` in the cost, it makes the throttle match the needed offset, reducing velocity error in steady state. Without `Ut_ref`, by just augmenting the Q value linked to the velocity tracking (the last value of Q), we observed a steady-state error in the order of $10^{-3} km/h$. However, with `Ut_ref` we could observe that this time the error was in the order of $10^{-10} km/h$, which could be qualified as offset free tracking (it is mainly a numerical error at that point). We can see the effect of `Ut_ref` in figure 6.8.



a Controller not using the `Ut_ref` cost addition



b Controller using `Ut_ref` in its cost

Fig. 6.8 | Velocity comparison graphs (without and with `Ut_ref`)

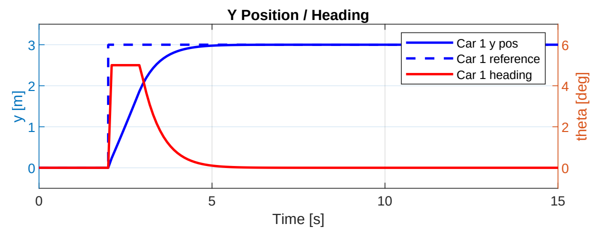
6.1.3 Performance of the controller

The results show that the desired lane change and velocity step are tracked smoothly, respecting all state and input limits. The lateral position y goes to 3 m with minimal overshoot, and the velocity converges from 80 to 100 km/h within the 5 s requirement. The steering δ and throttle u_T remain within $\pm 30°$ and ± 1.

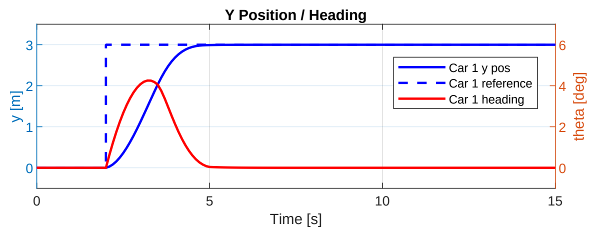
We noticed that the car was steering too abruptly at the beginning of lane changes, something that could not be resolved by simply increasing the R value related to steering, so we added a steering-difference constraint:

$$-0.001 \;\le\; \big(U(1,k) - U(1,k-1)\big) \;\le\; 0.001 \tag{6.35}$$

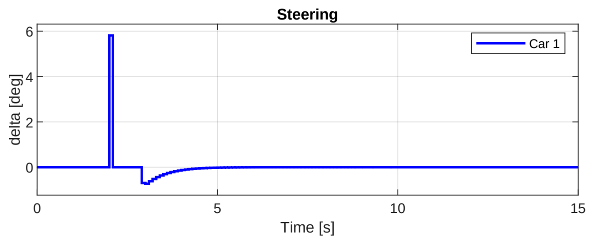
This makes the car avoid sharp changes in the steering δ, and smooths the lane transitions by limiting how quickly the steering input can vary over consecutive steps, as shown by comparing Figures 6.9c and 6.9d. The value of 0.001 was tuned until we could see a major difference at the beginning of the lane change. In Figure 6.9b, you can see how the y-position curve does not show any sharp angles as in Figure 6.9a, especially at the point where the car starts steering towards the overtaking lane (around the 2nd second), demonstrating the effect of this addition.



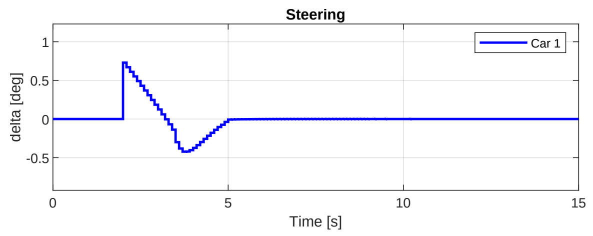
a y-position: Without the addition 6.35



b y-position: With the steering-rate constraint



c Steering: Without the addition 6.35



d Steering: With the steering-rate constraint

Fig. 6.9 | Comparison of y-position and steering behavior between controllers with and without the steering-rate constraint.

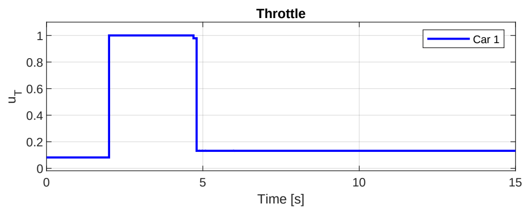


Fig. 6.10 | Final plot of the throttle

6.2 Design NMPC Controller for Overtaking (Deliverable 6.2)

6.2.1 Design procedure and choice of tuning parameters

We extend the tracking NMPC from Deliverable 6.1 by introducing $\mathbf{x}_{\text{other}}$, the state of the second (lead) car. In the code, $\mathbf{X}_{\text{other}}$ is propagated with a simplified assumption of constant speed of 80 km/h. Then, to prevent collisions, we add:

$$(\mathbf{X}_k - \mathbf{X}_{\text{other},k})^\top H_{\text{const}} (\mathbf{X}_k - \mathbf{X}_{\text{other},k}) \geq 1,$$

for some diagonal $H_{\text{const}} = \text{diag}\left(\frac{1}{d_x^2}, \frac{1}{d_y^2}\right)$. This defines an elliptical "keep-out zone" around the other car. The parameter d_y was set to 3m, corresponding to the lane width. For d_x, we found that a value of 15m gives a sufficient safety margin, making sure the ego car does not overtake too early or at the last second.

Cost Function Modifications. For overtaking, we changed the matrices Q and R to:

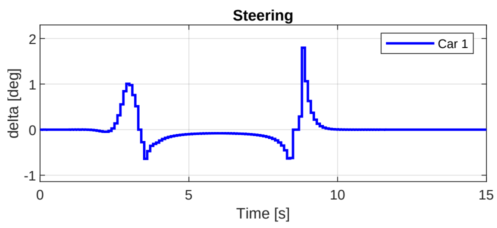
$$Q = \text{diag}(0,\ 1,\ 1,\ 300), \quad R = \text{diag}(60,\ 1),$$

giving more weight to speed tracking in Q (to avoid overshooting) and steering effort in R (for comfort). We also keep `Ut_ref` as in section 6.1.2 to handle steady-state speed error.

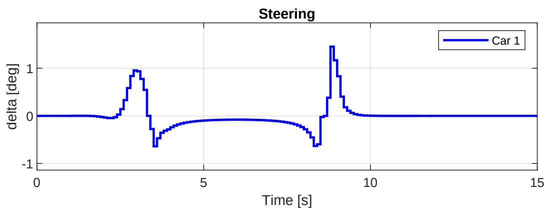
Steering-Rate Constraint. As in 6.1.3, we introduced a steering-rate constraint to avoid abrupt lateral motion during overtaking:

$$-0.01 \leq \big(U(1,k) - U(1,k-1)\big) \leq 0.01.$$

This constraint limits the rate of change of the steering angle between consecutive steps. Contrary to section 6.1.3, here we could have reached a similar result by increasing the R value linked to steering, but we still found better results by using our method, so we kept it. In 6.1.3 we used a stricter limit of 0.001; here we relaxed this constraint because we noticed the car starting to steer slightly to the right before initiating the left turn (trying to keep the strict steer-change constraint we gave it). The car is still very slightly turning to the right before turning left; you can see this where the y-position starts changing (around the 3rd second in the y-position in figure 6.13). You can see in figure 6.11 how the steering is kept at a lower level by using our additional constraint.



a Without the steering-rate constraint



b With the steering-rate constraint

Fig. 6.11 | Comparison of steering angle during the overtaking with and without the steering-rate constraint.

6.2.2 Performance of the controller

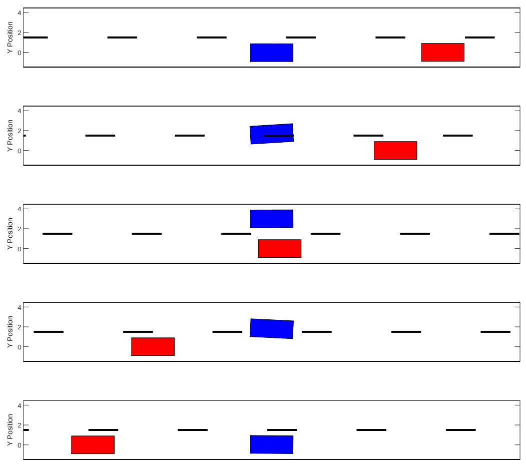


Fig. 6.12 | From top to bottom: the ego car (blue) overtakes the lead car (red) at higher speed, changing lanes.

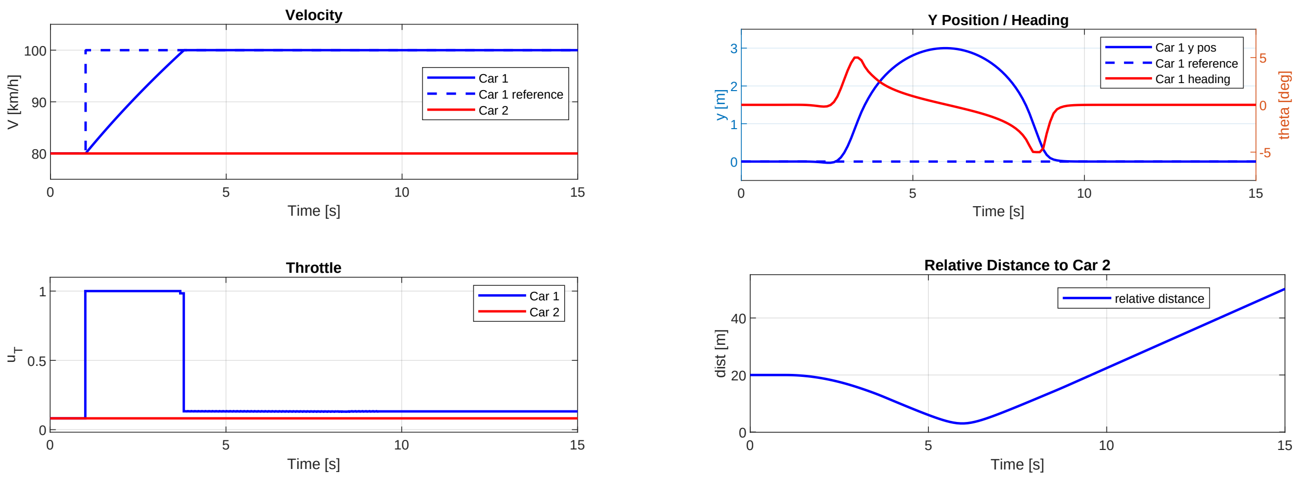


Fig. 6.13 | Final results of velocity, y-position, throttle, and relative distance during the overtaking maneuver.

With all these changes and tuning, the ego car performs the overtaking maneuver very smoothly. It leaves its original lane without abrupt steering, keeping the steering angle within $\pm 30°$. The car accelerates past the other vehicle while maintaining a safe distance and respecting the elliptical collision zone. When safely ahead, it returns to its initial lane.

The added steering difference constraint makes sure the maneuver is comfortable by avoiding sudden changes in direction. Additionally, the addition of the `Ut_ref` in the cost function helps the car maintain its desired speed after completing the overtaking maneuver, reducing steady-state speed errors.

7 Conclusion

We combined several approaches: linear MPC for separate longitudinal and lateral control, offset-free tracking to handle disturbances, robust tube MPC for adaptive cruise control, and a full nonlinear MPC for maneuvers like overtaking. Our results show that each method meets its own goals, like steady-state accuracy, safety under disturbances, or smooth lane changes. By tuning parameters and adjusting constraints, we were able to improve both performance and passenger comfort. The different MPC formulations work well together for controlling a car in various driving scenarios, from simple cruising to more difficult tasks like overtaking, and with more work and research we can certainly raise the bar higher in the future.