MPC - Final mini project

Groupe AD:
Nathann Morand (296190)
Felipe Ramirez (331471)
Tadej Strah (395623)

November 2024

Contents

Disclaimer

During this project, we divided the work equally, which led to some variations in the writing style throughout this report. These differences are particularly noticeable in the variable naming conventions within the code.

1 System dynamics and linearization

1.1 Derivations

We are tasked to derive the analytical expressions of A and B as a function of xs and us
Given the steady states:

$$x_s = (0, 0, 0, V_s), \quad u_s = (0, u_{T,s}),$$

and the system dynamics:

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

where:

$$\beta = \arctan\left(\frac{\ell_r \tan(\delta)}{\ell_r + \ell_f}\right),$$

$$F_{\text{motor}} = \frac{u_T P_{\max}}{V}, \quad F_{\text{drag}} = \frac{1}{2}\rho C_d A_f V^2, \quad F_{\text{roll}} = C_r m g.$$

At steady state:

$$x_s = \begin{bmatrix} 0 \\ 0 \\ 0 \\ V_s \end{bmatrix}, \quad u_s = \begin{bmatrix} 0 \\ u_{T,s} \end{bmatrix}.$$

Analytical Expressions for A

The linearization of the system around x_s involves finding the Jacobians:

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

$$\text{A} = \begin{bmatrix} 0 & 0 & -V_s\sin(\theta_s + \beta_s) & \cos(\theta_s + \beta_s) \\ 0 & 0 & V_s\cos(\theta_s + \beta_s) & \sin(\theta_s + \beta_s) \\ 0 & 0 & 0 & \frac{\sin(\beta_s)}{\ell_r} \\ 0 & 0 & 0 & -\frac{u_{T,s} P_{\max}}{m V_s^2} - \frac{\rho C_d A_f}{m} V_s \end{bmatrix}$$

At x_s:

$$\beta_s = 0, \quad \cos(\theta_s + \beta_s) = 1, \quad \sin(\theta_s + \beta_s) = 0,$$

thus we can simplify further

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

Analytical Expressions for B

The linearization of the system around u_s involves finding the Jacobians:

$$B = \left.\frac{\partial f(x, u)}{\partial u}\right|_{(x_s, u_s)} = \begin{bmatrix} 0 & 0 \\ \frac{l_r V_s}{l_r + l_f} & 0 \\ \frac{V_s}{l_r + l_f} & 0 \\ 0 & \frac{P_{\max}}{m V_s} \end{bmatrix}.$$

1.2 Deliverable 2.1

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & V_s & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{u_{T,s}P_{\max}}{mV_s^2} - \frac{\rho C_d A_f}{m} V_s \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ \frac{l_r V_s}{l_r + l_f} & 0 \\ \frac{V_s}{l_r + l_f} & 0 \\ 0 & \frac{P_{\max}}{mV_s} \end{bmatrix}.$$

1.3 Deliverable 2.2

In the A matrix, The only coupling between the longitudinal and lateral speed appears through the velocity Vs, which scales the lateral motion but does not affect the forces in the longitudinal direction.
The B matrix shows that throttle affects only the longitudinal dynamics and the steering angle affects only the lateral dynamics.
This is come from two things :

- **Independent controls:** The car's engine power affects its forward motion (longitudinal), while steering adjusts the car's orientation and lateral displacement.
- **Small-angle approximations:** In typical highway scenarios, steering angles are small, and the lateral motion does not significantly influence the forward motion.

2 MPC controllers for both subsystems

2.1 Deliverable 3.1

In this section, we designed a Model Predictive Control (MPC) controller that is both recursively feasible and stabilizing. This controller can track step references for the two subsystems defined in Part 2. These subsystems are as follows:

1. **Longitudinal subsystem**: characterized by the states $[x, v]$ and the input u_T, where v is the velocity and u_T is the throttle. In this subsystem, there are no constraints on the states x (longitudinal position) and v (velocity), making it unconstrained in terms of state dynamics. The only constraints apply to the input u_T, which is limited by the mechanical capacity of the throttle.
2. **Lateral subsystem**: characterized by the states $[y, \theta]$ and the input δ, where y represents the lateral position, θ the heading angle, and δ the steering angle. For this subsystem, constraints are imposed on both the states and the input to ensure the car remains within the lane and maintains user comfort during lane changes.

The system is subject to the following constraints:

- **State constraints (applied to the lateral subsystem only)**: Constraints are applied to the lateral position y and the heading angle θ to keep the car within the lane and ensure smooth transitions during lane changes. These constraints also help reduce linearization error:

$$-0.5\,\text{m} \le y \le 3.5\,\text{m}, \quad |\theta| \le 5^\circ = 0.0873\,\text{rad}.$$

- **Input constraints (applied to both subsystems)**: The input constraints ensure the throttle u_T in the longitudinal subsystem and the steering angle δ in the lateral subsystem remain within the physical limitations of the vehicle:

$$-1 \le u_T \le 1, \quad |\delta| \le 30^\circ = 0.5236\,\text{rad}.$$

To implement the two subsystems mentioned above, we used the following delta formulation:

$$\Delta\bar{x}_{k+1} = A\Delta\bar{x}_k + B\Delta u_k$$

$$J(\Delta\mathbf{x}) = \min \sum_{i=0}^{N-1} \left[\Delta\mathbf{x}_i^\top Q \Delta\mathbf{x}_i + \Delta\mathbf{u}_i^\top R \Delta\mathbf{u}_i\right] + \Delta\mathbf{x}_N^\top Q_f \Delta\mathbf{x}_N$$

where A and B are derived from the linearized model at the steady-state operating point (x_s, u_s). To ensure recursive feasibility, the input constraint $M \cdot u \leq m$ is enforced by properly shifting the delta formulation to account for u_s.

The cost function for the lateral subsystem includes penalties for deviations in y, θ, and δ, ensuring stability while respecting constraints. Using the principles of Linear Quadratic Regulator (LQR) control, we tuned Q, R, and Q_f. The terminal cost Q_f is derived from the discrete-time algebraic Riccati equation, ensuring the system remains within the terminal invariant set defined in the (y, θ) space. The closed-loop dynamics $A_{\text{cl}} = A + BK$ guarantee stability and constraint satisfaction.

2.2 Parameter tuning

The design of an efficient MPC controller required careful selection of control parameters (Q, R, H). Through systematic experimentation, we analyzed their impact on system performance and adjusted values to achieve optimal behavior.

- **Prediction horizon** (H): The prediction horizon balances the trade-off between accuracy and computational complexity. A shorter horizon limits the controller's ability to anticipate future states, while a longer horizon increases computational load. Extensive testing identified $H = 3$ seconds as providing a good balance.
- **Tuning** Q **and** R: A higher weight for Q caused aggressive control actions, particularly in throttle adjustments, while smaller Q ensured smoother responses. Conversely, increasing R improved control smoothness, reducing sudden shifts. We finalized the following parameter values:
  - Longitudinal subsystem:
  $$Q = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1$$
  - Lateral subsystem:
  $$Q = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}, \quad R = 1$$

The final parameter choices ensured smooth, stable control actions while respecting system constraints. The effectiveness of these settings was verified through simulations, and the results are detailed in the following figures:

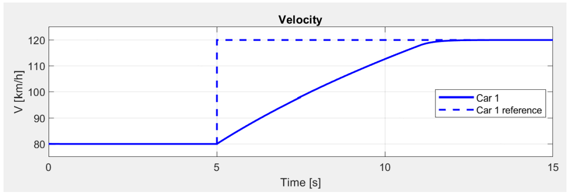

Figure 1: Velocity

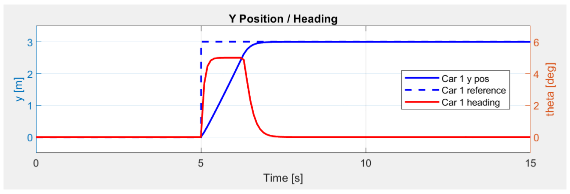

Figure 2: Y-position and θ orientation

Additional results from parameter tuning experiments for the lateral subsystem are shown below:

Thus, with the parameters decided above, we can determine the terminal invariant set of our lateral subsystem as follows:

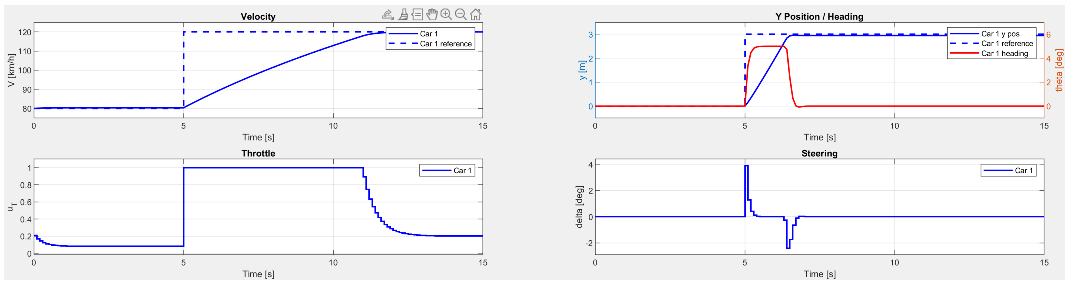

Figure 3: H = 1, Q = 1, R = 1

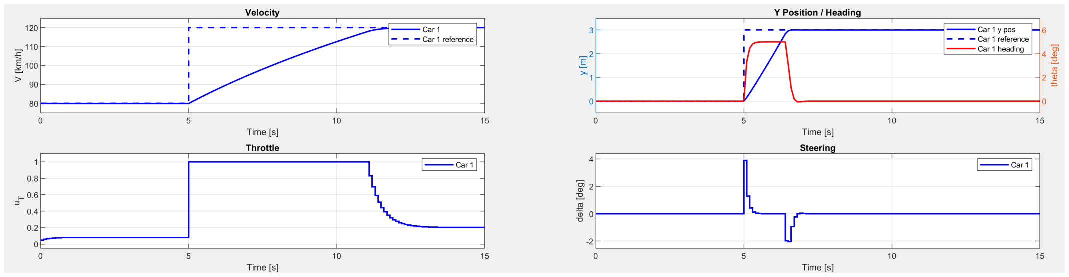

Figure 4: H = 6, Q = 50, R = 1

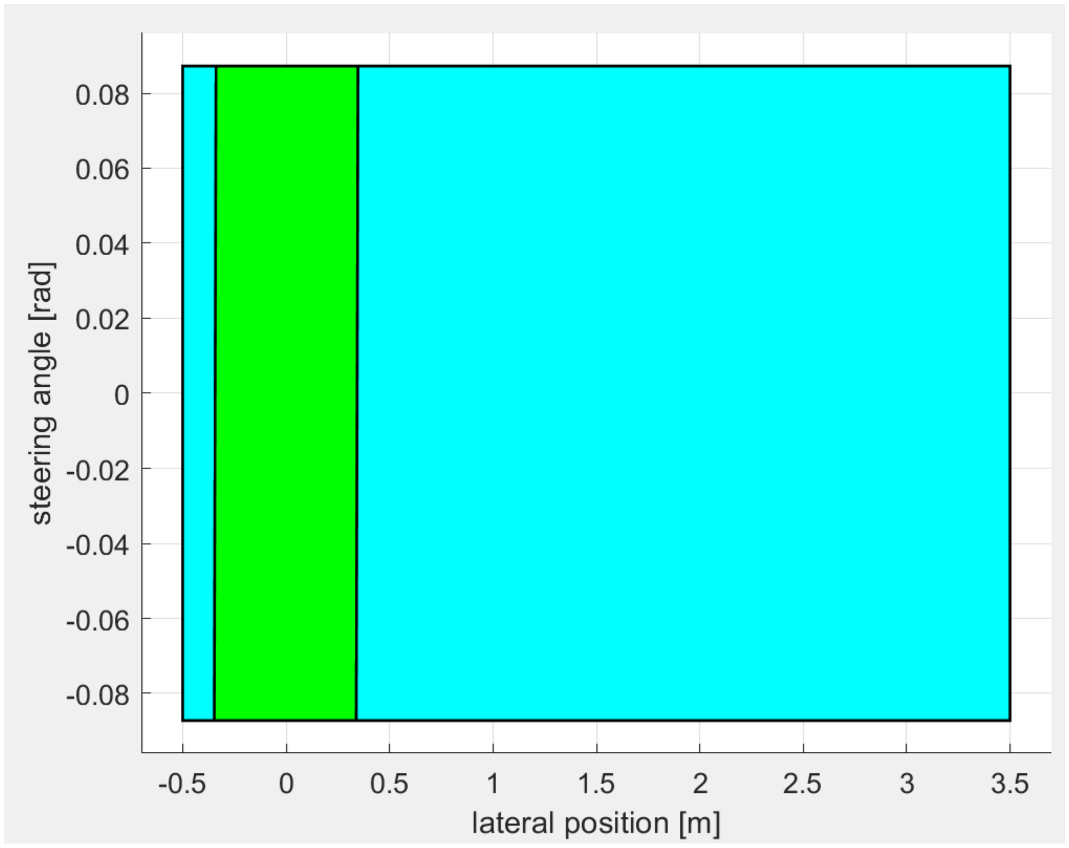

Figure 5: Terminal invariant set for the lateral sub-system

3 Offset-free tracking

3.1 Deliverable 4

The disturbance estimation in `LonEstimator` is achieved by introducing an extended state vector. This vector includes both the velocity V and the disturbance d, defined as:

$$\mathbf{z} = \begin{bmatrix} V \\ d \end{bmatrix}.$$

The system dynamics for this extended state are described by the following equation:

$$\dot{\mathbf{z}} = \hat{\mathbf{A}}\mathbf{z} + \hat{\mathbf{B}}u + \mathbf{L}\left(y - \hat{\mathbf{C}}\mathbf{z}\right),$$

where $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$, and $\hat{\mathbf{C}}$ are the augmented system matrices, and $\mathbf{L}$ is the observer gain matrix used to correct the estimation error.

In our design of the longitudinal disturbance estimator, the following matrices and equations were used:

1. **Augmented State Matrix:**

$$\mathbf{A}_{\text{hat}} = \begin{bmatrix} A_d(v,v) & B_d(v) \\ 0 & 1 \end{bmatrix}$$

2. **Augmented Input Matrix:**

$$\mathbf{B}_{\text{hat}} = \begin{bmatrix} B_d(v) \\ 0 \end{bmatrix}$$

3.**Augmented Output Matrix:**

$$\mathbf{C}_{\text{hat}} = \begin{bmatrix} C_d(v,v) & 0 \end{bmatrix}$$

4. **Observer Gain Matrix:**

$$\mathbf{L} = \texttt{place}(\hat{\mathbf{A}}^\top, \hat{\mathbf{C}}^\top, [0.5, 0.6])^\top$$

3.2 Parameters tuning

The poles were selected as 0.5 and 0.6 to ensure a balance between convergence speed and stability.

Furthermore, we reduced the time step T_s to 40 ms to account for the nonlinear dynamics, which previously introduced oscillations in the throttle.

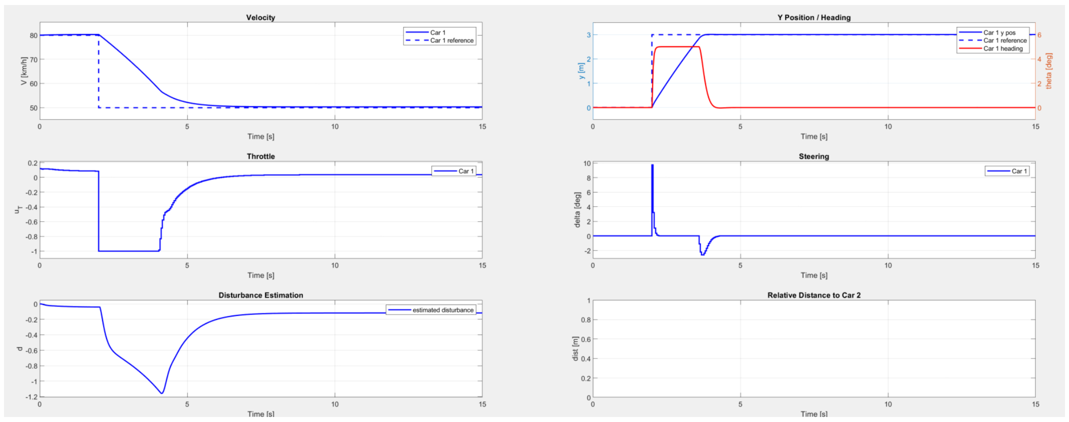

Figure 6: Velocity Tracking with Disturbance Estimation

4 Tube MPC

4.1 Deliverable 5.1

To design the tube controller, we have to derive the relative longitudinal dynamics:

$$\Delta = \begin{bmatrix} \Delta x \\ \Delta v \end{bmatrix}$$

and model the uncertainty caused by the lead car's throttle $\tilde{u}_T$ as a bounded disturbance.

A discrete-time linear quadratic regulator (LQR) controller K was computed using the matrices A_d and B_d obtained from the longitudinal subsystem.

4.2 Parameters tuning

The cost matrices:

$$Q = 10, 100, \quad R = 2$$

were chosen experimentally to balance responsiveness and smoothness.

The safe distance:

$$x_{\text{safe,pos}} = 8$$

was set to ensure a minimum bumper-to-bumper gap of 1.7 m, with a margin to allow for robust convergence within a 6-second horizon. These parameters provided stability and ensured the controller maintained the desired minimum gap under all allowable disturbances.

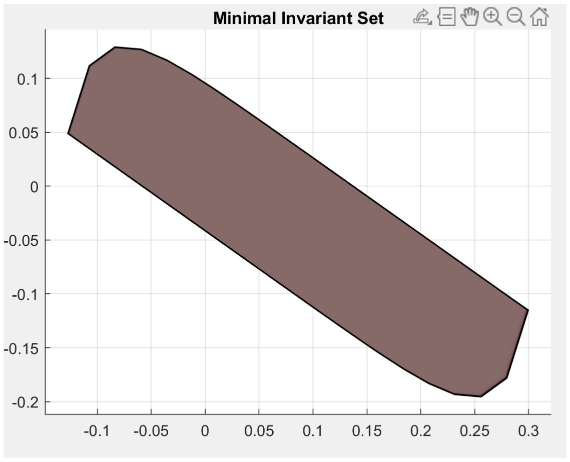

Figure 7: Minimal Invariant Set

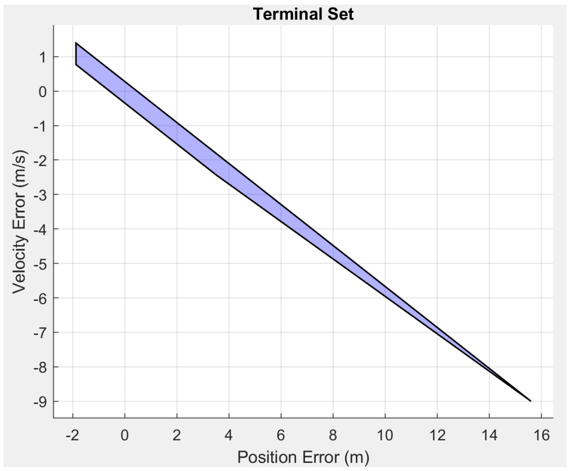

Figure 8: Terminal Set

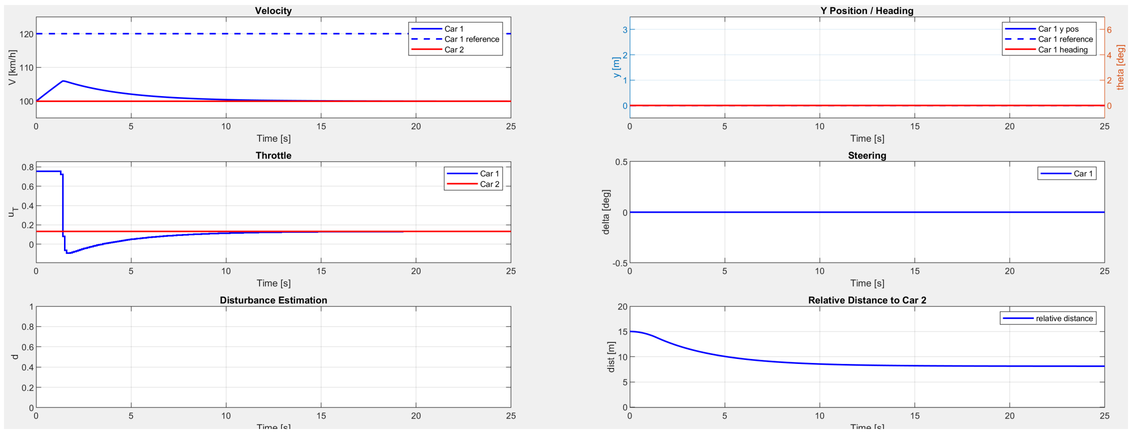

Figure 9: Robust controller simulation with `myCar.x0` $= [0, 0, 0, 100/3.6]'$ and `myCar.ref` $= [0, 120/3.6]'$. The other car starts at `otherCar.x0` $= [15, 0, 0, 100/3.6]'$ with `otherCar.u = car.u_const(100/3.6)`

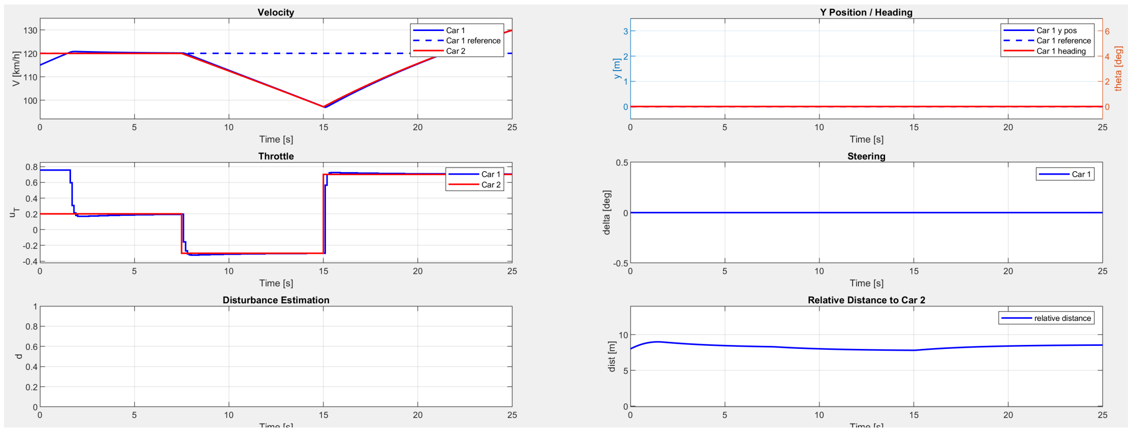

Figure 10: Robust controller simulation with `myCar.x0` $= [0, 0, 0, 115/3.6]'$ and `myCar.ref` $= [0, 120/3.6]'$. The other car starts at `otherCar.x0` $= [8, 0, 0, 120/3.6]'$ with `otherCar.u = car.u_fwd_ref()` and `otherCar.ref = car.ref_robust()`

5 Nonlinear MPC

5.1 Deliverable 6.1

In this part, we ditch the linearized and separated model of the car from before and work directly with the nonlinear model. To perform nonlinear optimization, we use the *Casadi* software package. We use the same state and input constraints as before, but this time we do not compute the terminal set.

Cost is computed as expected for a tracking problem; it is designed to minimize the velocity and lateral position error. Using equal weights of 1 for both of these variables proved to work well. To penalize excessive use of throttle and steering, again, weights of 1 for both inputs worked well enough. In matrix notation, we used

$$Q = \mathrm{diag}([0, 1, 0, 1]), \quad R = \mathrm{diag}([1, 1]).$$

We use the 4-th order Runge-Kutta method to discretize the system and obtain f_{discrete}. State evolution is then computed as $X_{k+1} = f_{\text{discrete}}(X_k, U_k)$.

We use the prediction horizon of $H = 2$ seconds, sampling time of , which resulted in stable control, but not too long computation times.

5.2 Steady-state error

Steady-state error arises from the linearization error. Since we're working directly with the nonlinear model there should be no steady-state error, as can be observed in 11. Velocity and lateral position of the car match the reference exactly (within numerical error).

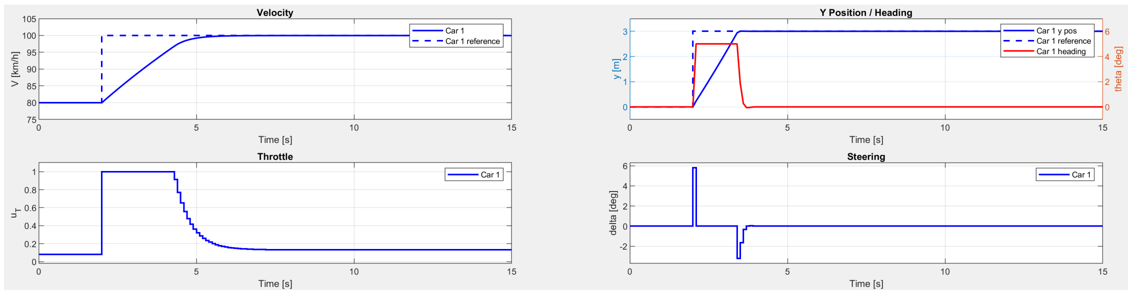

Figure 11: Performance of tracking NMPC controller

5.3 Deliverable 6.2

Controller for this part is mostly unchanged from the task 6.2, only that here we add a collision-avoidance constraint to avoid crashing into the other car.

We use a simple ellipsoidal constraint based on the physical distance between the centers of both cars. We define position vectors $p = [x, y]$ and $p_L = [x_l, y_l]$ for both cars and a diagonal weight matrix $H = \mathrm{diag}([H_{\mathrm{lon}}, H_{\mathrm{lat}}])$. The ellipsoidal constraint is therefore stated as

$$(p - p_l)^T H (p - p_l) \geq 1. \tag{1}$$

If we expand the equation 1, we get

$$H_{\mathrm{lon}}(x - x_l)^2 + H_{\mathrm{lat}}(y - y_l)^2 \geq 1. \tag{2}$$

5.4 Parameters tuning

The elements of the matrix H are related to the size of the collision avoidance ellipse as $H_{\mathrm{lon}} = 1/a^2$ and $H_{\mathrm{lat}} = 1/b^2$. A good start for choosing the ellipse axes a and b is to make them the size of a car. That would avoid the collision in case the cars were also the shape of ellipses, but as they are not, and to provide some extra safety distance we need to either choose a and b bigger. It the end we chose $H_{\mathrm{lon}} = 1/12^2$, $H_{\mathrm{lat}} = 1/3^2$ to ensure a safe overtaking maneuver.

We also had to rebalance the weights in the cost function. The weight on the velocity constraint had to be large enough ($Q_v = 300$) in relation the lateral position constraint ($Q_y = 1$), for the overtaking to even happen. The heading direction penalty was increased as well; without it, the car would initiate the overtake by slightly turning to the right, only to then take a sharper left turn. To remove this countersteer, a weight $Q_\theta = 100$ was used. Steering weight ($R_\delta = 200$) was also increased to prevent oscillations. Throttle use was also penalized quite heavily ($R_{u_t} = 100$) to ensure a smooth ride. We can see the final controller performance in 12.

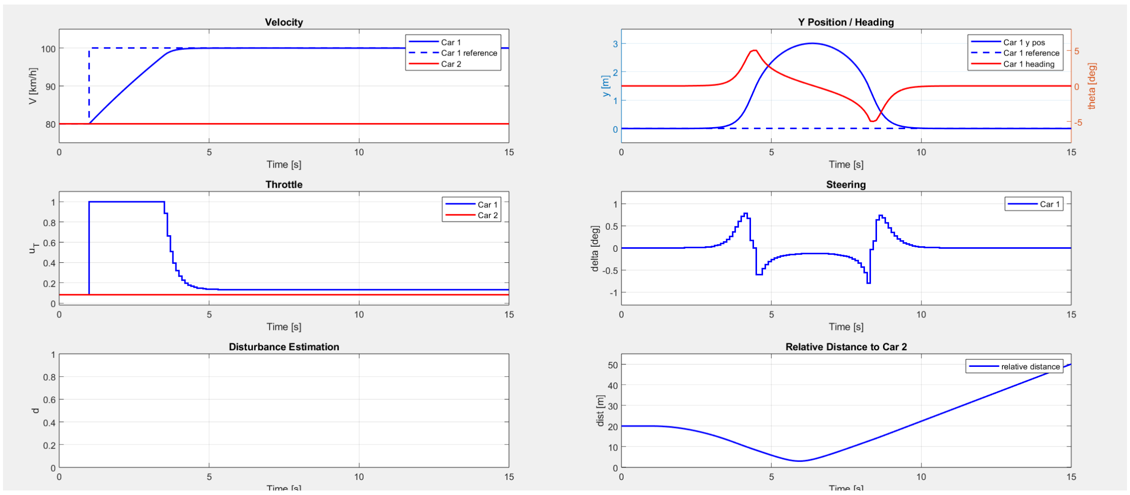

Figure 12: Performance of NMPC controller for overtaking