POLITECNICO DI TORINO
Repository ISTITUZIONALE

Autonomous Vehicle Trajectory Tracking: An Integrated Longitudinal/Lateral LPV MPC Approach

Original
Autonomous Vehicle Trajectory Tracking: An Integrated Longitudinal/Lateral LPV MPC Approach / Cerrito, F., Menezes Morato, M., Canale, M., Sename, O.. - (In corso di stampa). (23rd IFAC World Congress).

Availability:
This version is available at: 11583/3014107 since: 2026-08-07T07:19:17Z

Publisher:
Elsevier

Published
DOI:

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

11 September 2026

Autonomous Vehicle Trajectory Tracking: An Integrated Longitudinal/Lateral LPV MPC Approach ⋆,⋆⋆

F. Cerrito * **M. M. Morato** **,*** **M. Canale** * **O. Sename** ***

* *Politecnico di Torino, Torino 10129, Italy.*
** *Instituto Nacional de Ciência e Tecnologia em Controle e Automação de Processos de Energia (INCT CAPE), Univ. Fed. de Santa Catarina, Florianópolis-SC, Brazil.*
*** *Univ. Grenoble Alpes, CNRS, Grenoble INP*$^{\top}$*, GIPSA-lab, 38000 Grenoble, France.*
$\top$ *Institute of Engineering, Univ. Grenoble Alpes*

Abstract: The trajectory tracking problem for autonomous vehicles can be enhanced by means of an effective unification of longitudinal and lateral control. To this end, Nonlinear Model Predictive Control (NMPC) is highly suitable as it inherently handles multi-objetive problems and accounts for process constraints. However, a key challenge in implementing NMPC schemes lies in the involved computational burden, which can impede systems subject to stringent real-time constraints. This issue directly affects the optimization feasibility, its overall effectiveness, and closed-loop stability. To overcome these limitations, we employ a Linear Parameter-Varying (LPV) representation of the vehicle's dynamic model. This article proposes an integrated LPV MPC solution for autonomous vehicle trajectory tracking. Accordingly, we propose an integrated LPV MPC establishing guarantees for both recursive feasibility and closed-loop stability via the formulation of parameter-dependent Linear Matrix Inequalities. The proposed approach is validated through comparative simulations using a high-fidelity nonlinear dynamic model of a scaled vehicle, demonstrating similar tracking performances compared to NMPC but strongly reducing computation time, making it suitable for real time applications.

Keywords: Nonlinear and optimal automotive control, Linear parameter-varying systems, Model predictive control.

1. INTRODUCTION

Improving trajectory tracking performance in Autonomous Vehicles (AVs) represents an ongoing challenge in the field of autonomous systems (Sename et al., 2024). The primary objective of trajectory tracking control is to accurately follow a pre-planned, feasible reference trajectory while simultaneously ensuring that the involved input and state constraints are strictly satisfied, making minor trajectory adjustments when needed, e.g., for obstacle avoidance and a smoother ride, cf. (Chen et al., 2018). A considerable practical difficulty in this area arises due to the coupled nature of the vehicle's dynamics, as both longitudinal and lateral motion are intrinsically involved and must be managed concurrently. Furthermore, a pivotal requirement is that all control computations be performed online, ensuring both accurate tracking and consistent constraint satisfaction, while also guaranteeing that a valid control action is evaluated, cf. (Alcalá et al., 2019).

To effectively solve the path tracking problem, Corno et al. (2021) implement a Linear Parameter-Varying (LPV) Multiple-Input Single-Output $\mathcal{H}_\infty$ controller. This controller relies on feedback of the lateral error at the center of gravity and the look-ahead distance. Notably, the LPV framework is used here to account for the vehicle dynamics' dependence on the longitudinal velocity. Conversely, Duan et al. (2021) adopt a distinct lateral and longitudinal control decoupling strategy. In this approach, the lateral and heading error are managed by a Model Predictive Control (MPC) scheme, while the longitudinal motion is handled by combining a PID and a LQR controller.

The introduction of MPC for trajectory tracking is a common choice, particularly because it allows for the simultaneous management of multi-objective goals (e.g., tracking a reference speed and path) while explicitly accounting for plant constraints. For example, Reiter et al. (2023) propose a Nonlinear MPC (NMPC) formulation that employs different coordinate systems, incorporating cost and collision avoidance constraints. Similarly, Kloeser et al. (2020) utilize an NMPC approach based on a singularity-free prediction model, enabling the vehicle to operate across a broad range of conditions. Additional constraints are imposed in their optimal control problem to guarantee the validity of the underlying model assumptions.

Despite its capabilities, a primary limitation in the practical implementation of NMPC is the challenge in ensur-

⋆ This work is part of the project Piano Nazionale di Ripresa e Resilienza (PNRR)- Next Generation Europe, which has received funding from the Italian Ministry of University and Research – DM 117/2023.

This work has been funded by European Union's Horizon Europe research and innovation programme under the *Marie Skłodowska-Curie Action* grant agreement No 101149263 (*REAL OPT* 4 *CONTROL*). This work has been partially supported by ROBOTEX 2.0, the French Infrastructure in Robotics under the grants ROBOTEX (EQUIPEX ANR-10-EQPX-44-01) and TIRREX (EQUIPEX+ grant ANR-21-ESRE-0015).

⋆⋆An extended version of this article is available at `https://hal.science/hal-05394625`

ing the required online computational time, guaranteeing optimality, and rigorously proving controller stability and recursive feasibility. These are pivotal aspects for bridging the gap between simulation environments and real-world deployment. To address this, LPV systems (Shamma, 2012) have recently been employed successfully as the prediction model within MPC frameworks (Morato et al., 2020), offering a framework for computational efficiency and stability analysis.
In this article, we propose an integrated trajectory tracking strategy for AVs that leverages a simplified, control-oriented LPV model within a del MPC framework. The main contributions of this work are:

- *Control-Oriented LPV Modeling:* A derivation of a novel LPV model specifically tailored for integrated vehicle control purposes (Section 3).
- *LPV MPC formulation with guarantees:* A corresponding LPV MPC formulation that ensures closed-loop stability and recursive feasibility (Section 4). The necessary terminal ingredients are derived through adaptation of the results in Morato et al. (2023) and computed by solving the associated Linear Matrix Inequalities (LMIs).
- *Validation and benchmarking:* Realistic simulation results comparing the proposed scheme against state of the art NMPC benchmarks (Section 5). These results highlight the computational efficiency and enhanced performance enabled by our method.

Notation The symbol $\mathbb{N}$ denotes the set of non-negative integers. The set of integers over a specific interval $[0, a]$ as: $\mathbb{N}_a = \{i \in \mathbb{N} \mid 0 \leq i \leq a\}$. An array in $\mathbb{R}^{n\times 1}$ is denoted by a boldface lowercase letter $\mathbf{v}$. The standard MPC notation is used to denote predictions. The predicted value x at time instant $k+j$, computed using information available at the current time instant k, is denoted by $x(k+j|k)$.

2. PRELIMINARIES AND PROBLEM STATEMENT

The main goal of this work is to perform trajectory tracking while accounting for both longitudinal and lateral dynamics within a unified control problem. To achieve this, an LPV MPC approach is employed. To this end, the vehicle's dynamic model is suitably handled to first obtain a proper control-oriented model and, subsequently, to derive the necessary LPV prediction model.
As with any tracking control problem, the primary control argument is the reference trajectory itself. In the literature, numerous solutions have been proposed for trajectory planning (Guo et al., 2024), relying on various approaches such as optimization problems, learning-based methods, and function fitting. In this article, the existence of a feasible reference trajectory is assumed to be known a priori. The subsequent section details the specific features of this reference trajectory to establish a common foundation for the following technical developments. It is important to note that the assumptions and features detailed are commonly verified across a large variety of trajectory planners for AVs. Moreover, through trivial transformation, it is possible to represent a wide collection of existing trajectories within the proposed descriptive framework.
The reference trajectory $\mathbf{\Gamma}$ is defined in Cartesian coordinates as a vector function of time:

$$\mathbf{\Gamma}(t) = \mathbf{p}(t) = \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} . \tag{1}$$

Given the definition in (1), the following fundamental geometric and kinematic quantities are derived:

- *Curvilinear abscissa*: the arc length $s(t)$ from the starting point along the trajectory is defined by the integral of the coordinate speed:

$$s(t) = \int_0^t \sqrt{(\dot{x}_r(\tau))^2 + (\dot{y}_r(\tau))^2} d\tau \, .$$

- *Speed profile*: the instantaneous reference speed $v_r(t)$ can be computed as the norm of the derivative of the position vector, i.e.:

$$v_r(t) = ||\dot{\mathbf{p}}(t)||_2 = \sqrt{\dot{x}_r(t)^2 + \dot{y}_r(t)^2} \, .$$

- *Reference heading angle*: the reference heading angle $\psi_r(t)$ is the angle of the tangent to the trajectory:

$$\psi_r(t) = \operatorname{atan}\left(\dot{y}_r(t), \dot{x}_r(t)\right) \, .$$

- *Trajectory curvature*: the rate of change of the heading angle $\kappa(s)$ with respect to the curvilinear abscissa, that is equal to the reciprocal of the radius of curvature $R(s)$:

$$\kappa(s) = \frac{d\psi_r}{ds} = \frac{1}{R(s)} \, .$$

Assumption 1. Consider a given reference trajectory Γ given in the Cartesian format of Eq. (1). Then, the following requirements hold:

- *Curvature smoothness*: the trajectory curvature $\kappa(s)$ is twice differentiable with respect to s, i.e., $\kappa(s) \in \mathcal{C}^2$.
- *Trackable speed profile*: the reference speed profile $v_r(t)$ is dynamically feasible and trackable by the vehicle while satisfying all actuator constraints. That is, considering the vehicle's acceleration limits, the speed can indeed be imposed by some well-posed control action.
- *Trackable trajectory curvature*: the trajectory curvature $\kappa(s)$ is dynamically feasible and trackable by the vehicle, satisfying all actuation constraints (e.g., limits on steering wheel speed). Crucially, the vehicle's steering system imposes a geometric limit such that the steering wheel angle δ remains inherently small.

3. LPV VEHICLE MODEL

This Section begins with the presentation of the plant model used in this study. Subsequently, the model is transformed using Frenet-Serret coordinates, and the resulting advantages for trajectory tracking are discussed. Finally, we derive a control-oriented LPV representation of the system, later used for the NMPC synthesis.

3.1 Cartesian model

For control purposes, a Single Track Kinematic (STK) model is employed (Fig. 1). Although this model does not capture the full vehicle dynamics, it is well known in the literature that this representation is reliable for vehicle operating at non-high-speed, cf. Kong et al. (2015).
The dynamic of the vehicle for $t \in \mathbb{R}_{\geq 0}$, in Cartesian coordinates, is described in (2) by the vehicle position (x, y), orientation ψ, actual speed v, and steering angle δ. The vehicle is controlled by the longitudinal acceleration a and wheel steering rate w_δ. The parameter L represents the vehicle wheelbase (i.e., the distance between the front and rear wheel axles).

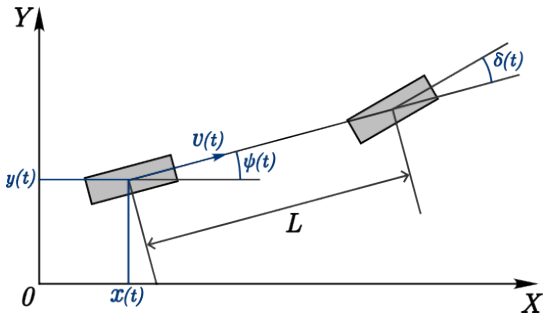

Fig. 1. STK Cartesian reference frame.

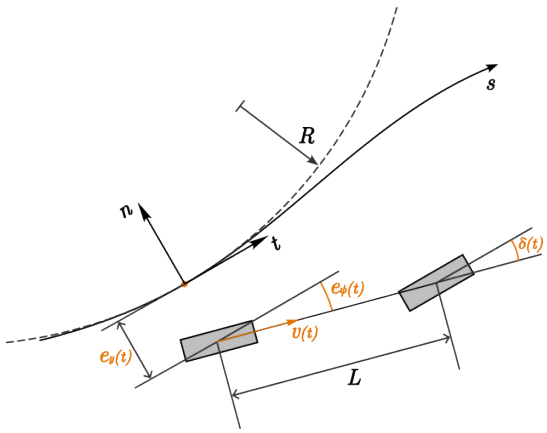

Fig. 2. STK Frenet reference frame

$$\begin{cases} \dot{x}(t) = v(t)\cos(\psi(t)) \\ \dot{y}(t) = v(t)\sin(\psi(t)) \\ \dot{\psi}(t) = \dfrac{v(t)}{L}\tan(\delta(t)) \\ \dot{v}(t) = a(t) \\ \dot{\delta}(t) = \omega_\delta(t) \end{cases} . \tag{2}$$

3.2 Frenet model

Although the model in (2) properly captures the plant dynamics, its state formulation presents notable limitations for control design and trajectory tracking.
Specifically, formulating the control problem in terms of a standard tracking error objective is challenging. Furthermore, the vehicle position states (x, y) are tightly coupled by the trigonometric relationship defined by the vehicle's orientation ψ, introducing complexity. Finally, the model does not explicitly incorporate essential geometric information about the reference trajectory's curvature, which is crucial for efficient path following. To address the aforementioned limitations, considering that Assumption 1 holds for our problem, we replace the model based on the usual Cartesian representation by a new one, which is based on Frenet-Serret coordinates, as illustrated in Fig. 2. The local Frenet frame is defined by the unit tangent vector $\mathbf{t}(s)$ and unit normal vector $\mathbf{n}(s)$ of the reference trajectory $\mathbf{\Gamma}$ at the generic curvilinear abscissa s. Let $\bar{s}$ denote the curvilinear abscissa related to the reference trajectory $\Gamma(t)$ in (1) corresponding to the orthogonal projection of the actual vehicle position onto the reference trajectory (i.e., the point of minimum Euclidean distance). The vehicle's Cartesian position vector, $[x, y]^T$, is then expressed as a function of the reference point $\mathbf{r}(\bar{s}) = [x_r(\bar{s}), y_r(\bar{s})]^T$ plus the lateral deviation e_y along the normal vector $\mathbf{n}(\bar{s})$:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{r}(\bar{s}) + e_y\mathbf{n}(\bar{s}) = \begin{bmatrix} x_r(\bar{s}) - e_y\sin(\psi_r(\bar{s})) \\ y_r(\bar{s}) + e_y\cos(\psi_r(\bar{s})) \end{bmatrix} . \tag{3}$$

By defining the vehicle heading error e_ψ as the difference between the actual vehicle heading ψ and the reference path heading angle at $\bar{s}$, the actual vehicle heading angle can be expressed as:

$$\psi = \psi_r(\bar{s}) + e_\psi . \tag{4}$$

It is worth noting that in equations (3) and (4), the vehicle's state is explicitly expressed as a function of the reference trajectory. This transparently highlights the tracking problem, as the vehicle's state is represented by the lateral error (e_y) and the heading error (e_ψ).
By performing a suitable time-derivative transformation (e.g. applying the chain rule and the Frenet-Serret formulas), the following equivalent dynamic model in Frenet coordinates is obtained:

$$\begin{cases} \dot{s}(t) = v(t)\dfrac{\cos(e_\psi(t))}{1-\kappa(s)e_y(t)} \\ \dot{e_y}(t) = v(t)\sin(e_\psi(t)) \\ \dot{e_\psi}(t) = v(t)\left[\dfrac{1}{L}\tan(\delta(t)) - \kappa(s)\dfrac{\cos(e_\psi(t))}{1-\kappa(s)e_y(t)}\right] \\ \dot{v}(t) = a(t) \\ \dot{\delta}(t) = \omega_\delta(t) \end{cases} . \tag{5}$$

Assumption 2. The Frenet-frame transformation is assumed to be valid if the vehicle remains sufficiently close to the reference path such that $e_y(t)$ and $e_\psi(t)$ are small. Consequently $1-\kappa(s(t))\,e_y(t) > 0$. These conditions ensure the uniqueness of the orthogonal projection onto the path and prevent singularities in the Frenet-coordinate model.

The obtained representation allows for describing the vehicle dynamics directly as a function of the tracking errors (e_y and e_ψ), enabling a straightforward control problem design. In addition, the information about the reference trajectory's curvature is natively accounted for in the model.
We emphasize that, despite the fact that the full model in (5) includes states related to the curvilinear axis (s), this variable is not actually the primary focus for trajectory tracking. Indeed, appropriate trajectory control can be ensured by regulating the vehicle's longitudinal speed to some reference value while, simultaneously, steering the lateral and heading errors to the origin (lateral control).

3.3 Control-oriented LPV model

A major limitation of implementing MPC for nonlinear systems is the challenge of solving the online optimization problem efficiently to ensure both proper computational time and optimality (see Morato et al. (2024)). In order to avoid such numerical difficulties, considering the strict real-time constraints characteristics of autonomous vehicles, we employ an LPV representation of the system description from (5), in the fashion of (Morato et al., 2020). Due to the physical limitations of the steering system introduced in Assumption 1, the vehicle steering wheel angle (δ) is restricted to small values, justifying the approximation:

$$\delta \approx \tan(\delta) . \tag{6}$$

Similarly, a linear approximation is applied to the heading error (e_ψ). Under Assumption 2, guaranteeing the validity of the Frenet transformation, the small-angle approximation holds:

$$e_\psi \approx \sin(e_\psi) . \tag{7}$$

To complete the transformation into an LPV framework, the following scheduling parameters are introduced:

$$\rho_1(t) = v(t)\,, \tag{8}$$

$$\rho_2(t) = \kappa(s)\frac{\cos\left(e_\psi(t)\right)}{1-\kappa(s)e_y(t)}\,. \tag{9}$$

With respect to the scheduling terms defined in (8)-(9), it is noted that the first parameter ρ_1, directly corresponds to the vehicle speed, whereas the second parameter ρ_2, captures the influence of the trajectory curvature and the lateral deviation on the dynamic evolution of the heading error. Both scheduling parameters are functions of known and measured vehicle states. Furthermore, under Assumption 2, both parameters are guaranteed to be bounded.
Considering the selected scheduling parameters and assuming that Assumptions 1, 2 holds (thereby ensuring the validity of (8) and (9)), an exact LPV inclusion is applied (cf. Shamma (2012)). Consequently, the model in (5) can be reformulated in the following LPV form:

$$\begin{cases} \dot{e_y}(t) = \rho_1(t)e_\psi(t) \\ \dot{e_\psi}(t) = \dfrac{\rho_1(t)}{L}\delta(t) - \rho_2(t)v(t) \\ \dot{v}(t) = a(t) \\ \dot{\delta}(t) = \omega_\delta(t) \end{cases}\,. \tag{10}$$

Finally, defining the state vector (11) and the discretization time T_s, by a suitable discretization method, the discrete parameter-affine model (10) is obtained.

$$\mathbf{x} = [e_y \;\; e_\psi \;\; v \;\; \delta]^T \tag{11}$$

$$\mathbf{x}(k+1) = A_\rho(k)\mathbf{x}(k) + Bu(k)\,,$$
$$A_\rho(k) = A_0 + A_1\rho_1(k) + A_2\rho_2(k)\,,$$

$$A_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\,,\quad A_1 = \begin{bmatrix} 0 & T_s & 0 & 0 \\ 0 & 0 & 0 & \dfrac{T_s}{L} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\,,$$

$$A_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -T_s & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\,,\qquad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}\,.$$

4. LPV MPC CONTROLLER

Considering the LPV-oriented vehicle model presented in Section 3, this Section begins by detailing the necessary state and input rescaling of the plant model (10) to conditioning the optimization problem. Hence, the MPC problem leveraging the LPV formulation is formally stated.

4.1 Scaled system

To properly formulate the MPC optimization problem, we first re-scale (normalize) the involved state variable, in order to avoid numerical issues in the optimization problems (for the MPC implementation and also for the synthesis of required stabilizing ingredients using LMIs). This process is also necessary in order to ensure that all states, along with their respective measurement units, possess homogeneous magnitudes (i.e., they are of a comparable numerical scale).
For this purpose, a diagonal scaling matrix S is introduced. Each diagonal element of S is used to rescale the corresponding state variable appropriately. Accordingly, we define the following state transformation:

$$\mathbf{x} \doteq S\mathbf{z} \quad \mathbf{z} = S^{-1}\mathbf{x}\,. \tag{12}$$

where $\mathbf{x}$ is the unscaled state vector and $\mathbf{z}$ is the scaled state vector, the equivalent system dynamics from LPV representation in (10) for MPC prediction is defined as:

$$\mathbf{z}(k+1) = S^{-1}A_\rho S\mathbf{z}(k) + S^{-1}B\mathbf{u}(k)\,. \tag{13}$$

Consequently, the scaled discrete-time model becomes:

$$\mathbf{z}(k+1) = \; A_z\mathbf{z}(k) + B_z\mathbf{u}(k)\,. \tag{14}$$

where $A_z = S^{-1}A_\rho S$ and $B_z = S^{-1}B$.
This same transformation applies to the definition of the system state constraints and references. Specifically, given the minimum, maximum, and reference state vectors ($\mathbf{x}_{\min}$, $\mathbf{x}_{\max}$, and $\mathbf{x}_{\text{ref}}$) in the original unscaled system, the equivalent constraints and reference are defined as follow:

$$\mathbf{x}_{\min} \le \mathbf{x} \le \mathbf{x}_{\max} \Rightarrow S^{-1}\mathbf{x}_{\min} \le \mathbf{z} \le S^{-1}\mathbf{x}_{\max}\,, \tag{15}$$
$$\mathbf{z}_{\min} = S^{-1}\mathbf{x}_{\min}\;,\; \mathbf{z}_{\max} = S^{-1}\mathbf{x}_{\max}\;,\; \mathbf{z}_{\text{ref}} \;= S^{-1}\mathbf{x}_{\text{ref}}\,.$$

4.2 NMPC scheme

The tracking error vector $\mathbf{e}$, at the generic time instant k, is defined as the difference between the actual scaled state $\mathbf{z}(k)$ and its reference $\mathbf{z}_{\text{ref}}(k)$:

$$\mathbf{e}(k) = \mathbf{z}(k) - \mathbf{z}_{\text{ref}}(k)\,. \tag{16}$$

It is important to note that, given the system's Frenet-based state representation (10), the reference values for the first two states (lateral error e_y and heading error e_ψ) are always zero. This condition is inherently preserved in the scaled system due to (12).
The optimal control input $\mathbf{u}^\star$ at the current time instant k is determined by the resolution of (17). The OCP is constrained by ensuring adherence to the scaled LPV prediction model (10) over the prediction horizon N_p. The tracking error $\mathbf{e}(k)$ is contained within minimum and maximum bounds. At the same time, the control input and its rate of change are also restricted to account for the actuator's physical limitations. Finally, closed-loop stability is enforced through a terminal set constraint, which ensures that the error state at the end of the prediction horizon remains within a specific ellipsoidal region defined by the parameter-dependent matrix P_ρ.

$$\mathbf{U}^\star(k) = \arg\min_{\mathbf{U(k)}} J(U(k)) \tag{17}$$

subject to:

$$\begin{aligned} &\mathbf{z}(k+j+1|k) = A_\rho(k+j|k)\mathbf{z}(k+j|k)\,, && \forall k \in \mathbb{N}_{N_p-1}\,, \\ &\qquad\qquad + B\mathbf{u}(k+j|k)\,, && \\ &|\mathbf{e}(k+j|k)| \le \mathbf{e}_{\max}, && \forall k \in \mathbb{N}_{N_p}\,, \\ &|\mathbf{u}(k+j|k)| \le \mathbf{u}_{\max}, && \forall k \in \mathbb{N}_{N_p-1}\,, \\ &|\Delta\mathbf{u}(k+j|k)| \le \Delta\mathbf{u}_{\max}, && \forall k \in \mathbb{N}_{N_p-2}\,, \\ &\mathbf{e}(k+H_p|k)P_\rho\mathbf{e}(k+H_p|k) < 1\,. && \end{aligned}$$

The cost function $J\left(\mathbf{U}(k)\right)$ (18) is defined by the optimal control sequence $\mathbf{U}(k)$ (19) over the finite prediction horizon N_p. This objective function is structured as a summation of two parts: the stage cost $\ell(k)$ (20) that accounts for controller tracking performances using the state and input weighting matrices (Q and R), and a terminal cost term (21) defined by the parameter-dependent matrix P_ρ.

Parameter	Value
T_s	0.1 s
H_p	10
S	[0.01 1 0.01 1]
Q	[10 10 0.1 1]
R	[50 1]
α	0.95
$\mathbf{e}_{\max}$	[0.1 0.5 0.1 0.1]
$\mathbf{u}_{\max}$	[0.1 0.3]

Table 1. Controller parameters

$$J\left(\mathbf{U}(k)\right) = \sum_{k=0}^{N_p-1} \ell(k) + V_\rho(N_p)\,, \tag{18}$$

$$\mathbf{U}(k) = \left[\mathbf{u}(k)^T, \ldots, \mathbf{u}(N_p-1)^T\right]^T\,, \tag{19}$$

$$\ell(k) = \mathbf{e}(k+j|k)^T Q \mathbf{e}(k+j|k)\,, \tag{20}$$
$$+\mathbf{u}(k+j|k)^T R \mathbf{u}(k+j|k)\,,$$

$$V_\rho(N_p) = \mathbf{e}(k+N_p|k)^T P_\rho \mathbf{e}(k+N_p|k)\,. \tag{21}$$

To ensure adherence between the predicted scheduling parameters and the actual system values, the following manipulations are performed, which are valid under the assumption of good tracking performance (as guaranteed by the MPC constraints and Assumption 1).

- The first scheduling parameter ρ_1, is predicted by substituting the predicted vehicle speed $v(k+j|k)$ with the reference vehicle speed v_r, at the predicted time step:

$$\rho_1(k+j|k) = v_r(k+j|k) \tag{22}$$

- In the second scheduling parameter ρ_2, the trajectory curvature κ is predicted, while the tracking errors (e_y and e_ψ) are frozen at the current time step k. This simplification is justified by the fact that the tracking errors are expected to remain small due to the MPC constraint.

$$\rho_2(k+j|k) = \kappa(k+j|k)\frac{\cos(e_\psi(k))}{1-\kappa(k+j|k)e_y(k)} \tag{23}$$

To ensure proper control performance and recursive feasibility, it is necessary to guarantee that the combination of the reference trajectory and the maximum allowable tracking error remains strictly within the defined physical plant constraints. Specifically, the limits on the tracking error and the reference state must be consistent with the overall state bounds for all prediction steps.

$$\max(|\mathbf{z}_{\text{ref}}(k)|) + \mathbf{e}_{\max} \leq \mathbf{z}_{\max} \tag{24}$$

$$\min(|\mathbf{z}_{\text{ref}}(k)|) - \mathbf{e}_{\max} \geq \mathbf{z}_{\min} \tag{25}$$

5. SIMULATION

The proposed approach is validated through realistic simulations performed within the MATLAB/Simulink environment. A high-fidelity model of the Grenoble GIPSA-lab scaled vehicle prototype is used as the plant dynamics model (cf. Borrell et al. (2024)).
The proposed LPV MPC controller, detailed in Section 4, is implemented using the high-performance QP solver MOSEK for the online optimization, utilizing the parameters reported in Table 1. The terminal ingredients for stability and recursive feasibility are obtained by adapting the results in Morato et al. (2023).
For performance evaluation, the proposed LPV MPC approach is benchmarked against two distinct implementations to ensure a comprehensive comparative analysis.

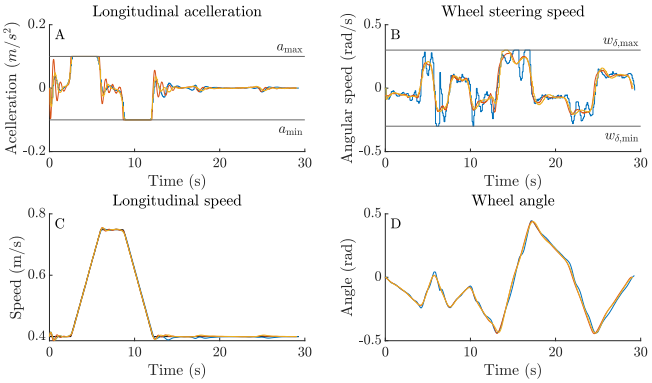

Fig. 3. — LPV MPC, — MATLAB, — CasADi.
(A) Longitudinal acceleration (a), (B) Wheel angle speed (w_δ), (C) Vehicle longitudinal speed (v_x), and (D) Wheel angle (δ).

The first benchmark is a standard NMPC solution implemented using the MATLAB Model Predictive Control Toolbox. The second benchmark is an NMPC implemented by CasADi utilizing the efficient IPOPT solver. The NMPC utilizes the discretized model (2) for prediction and shares the same plant simulated model, constraints, and reference trajectory as the LPV MPC controller.
The reference trajectory fulfills the smoothness and feasibility requirements described in Assumption 1. It is designed to be challenging, incorporating both speed profile variations and diverse turning maneuvers to fully evaluate the controller's performance under dynamic conditions.
As depicted in Fig.3, the LPV MPC and NMPC controllers achieve a highly equivalent performance across both actuation and resulting vehicle states. The longitudinal control actions (acceleration, Fig.3A) follow a similar trend, ensuring that the vehicle speed (Fig.3C) accurately tracks the reference profile in both scenarios. Similarly, the lateral control input w_δ generates comparable results of steering wheel angle (δ, Fig.3B and D), effectively demonstrating that the LPV approximation ensures good fidelity when managing the coupled dynamics of the plant.
Fig. 4 summarizes the comparative performance of the LPV MPC and the NMPC controllers. The tracking accuracy confirms the suitability of the LPV approach: the LPV MPC controller demonstrates similar performance in lateral error (Fig.4A), achieving a maximum lateral deviation that is slightly higher than, but comparable to, the NMPC across the full simulation run. Similarly, the heading error (Fig.4B) is comparable between both methods, consistently remaining below 0.1 rad. These small errors in both lateral and heading dynamics imply a good overall trajectory tracking, as visually confirmed by the superposition of the simulated trajectories with the reference one (Fig.5A).
The primary performance distinction lies in computational efficiency. Fig. 5B compares the optimization time, revealing a considerably noticeable difference: the NMPC implementations require a significantly longer optimization time per step, with an average time of 109 ms for the MATLAB implementation and 28 ms in CasADi. In contrast, the LPV MPC approach achieves real-time capability with an average optimization time of just 4 ms. Furthermore, it is worth noting that no formal terminal stability conditions, which are guaranteed by the LPV MPC implementation, were added to the NMPC benchmarks. This difference represents a computational speedup of nearly one order of magnitude, demonstrating the efficiency and practical

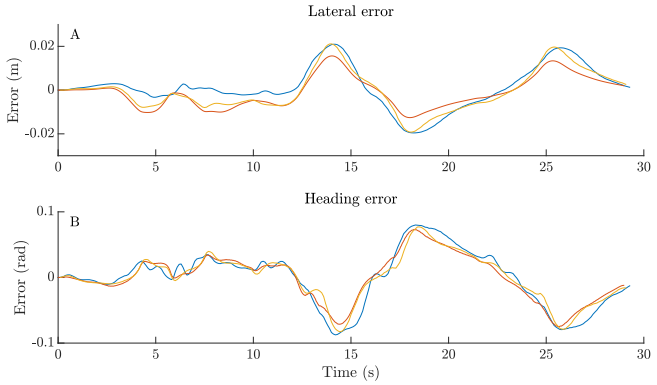

Fig. 4. — LPV MPC, — MATLAB, — CasADi. (A) Lateral error (e_y), and (B) Heading error (e_ψ)

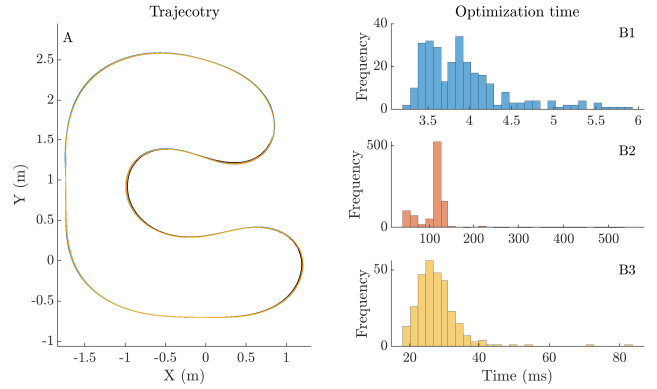

Fig. 5. — LPV MPC, — MATLAB, — CasADi. (A) Vehicle trajectory, and (B) Optimization time.

viability of the LPV formulation for fast online optimal control.

6. CONCLUSION

This paper presented an integrated LPV MPC framework designed to address the trajectory tracking problem for AVs. The primary motivation was to develop a controller that retains the multi-objective, constrained optimization benefits of NMPC while overcoming its significant computational burden, which is a common barrier to real-time implementation. The proposed solution was validated via comparative simulations against two NMPC benchmarks. The results show that the LPV MPC achieves performance comparable to NMPC, with tracking errors of the same order of magnitude. Notably, the LPV MPC achieves a significantly improved computational performance, with an average optimization time of 4 ms, corresponding to nearly one order of magnitude speedup compared to a CasADi-based NMPC. This real-time capability is achieved while preserving formal guarantees of recursive feasibility and closed-loop stability via terminal ingredients, which are not enforced in the NMPC benchmarks.

REFERENCES

Alcalá, E., Puig, V., and Quevedo, J. (2019). Lpv-mpc control for autonomous vehicles. *IFAC-PapersOnLine*, 52(28), 106–113. doi:https://doi.org/10.1016/j.ifacol.2019.12.356. 3rd IFAC Workshop on Linear Parameter Varying Systems LPVS 2019.

Borrell, A.M., Puig, V., and Sename, O. (2024). Fixed-structure parameter-dependent state feedback controller: A scaled autonomous vehicle path-tracking application. *Control Engineering Practice*, 147, 105911. doi:https://doi.org/10.1016/j.conengprac.2024.105911.

Chen, Y., Peng, H., and Grizzle, J. (2018). Obstacle avoidance for low-speed autonomous vehicles with barrier function. *IEEE Transactions on Control Systems Technology*, 26(1), 194–206. doi:10.1109/TCST.2017.2654063.

Corno, M., Panzani, G., Roselli, F., Giorelli, M., Azzolini, D., and Savaresi, S.M. (2021). An lpv approach to autonomous vehicle path tracking in the presence of steering actuation nonlinearities. *IEEE Transactions on Control Systems Technology*, 29(4), 1766–1774. doi:10.1109/TCST.2020.3006123.

Duan, X., Wang, Q., Tian, D., Zhou, J., Wang, J., Sheng, Z., Zhao, D., and Sun, Y. (2021). Implementing trajectory tracking control algorithm for autonomous vehicles. In *2021 IEEE International Conference on Unmanned Systems (ICUS)*, 947–953. doi:10.1109/ICUS52573.2021.9641133.

Guo, Y., Guo, Z., Wang, Y., Yao, D., Li, B., and Li, L. (2024). A survey of trajectory planning methods for autonomous driving—part i: Unstructured scenarios. *IEEE Transactions on Intelligent Vehicles*, 9(9), 5407–5434. doi:10.1109/TIV.2023.3337318.

Kloeser, D., Schoels, T., Sartor, T., Zanelli, A., Prison, G., and Diehl, M. (2020). Nmpc for racing using a singularity-free path-parametric model with obstacle avoidance. *IFAC-PapersOnLine*, 53(2), 14324–14329. doi:https://doi.org/10.1016/j.ifacol.2020.12.1376. 21st IFAC World Congress.

Kong, J., Pfeiffer, M., Schildbach, G., and Borrelli, F. (2015). Kinematic and dynamic vehicle models for autonomous driving control design. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, 1094–1099. doi:https://doi.org/10.1109/IVS.2015.7225830.

Morato, M.M., Cunha, V.M., Santos, T.L., Normey-Rico, J.E., and Sename, O. (2024). A robust nonlinear tracking mpc using qlpv embedding and zonotopic uncertainty propagation. *Journal of the Franklin Institute*, 361(6), 106713. doi:https://doi.org/10.1016/j.jfranklin.2024.106713.

Morato, M.M., Holicki, T., and Scherer, C.W. (2023). Stabilizing model predictive control synthesis using integral quadratic constraints and full-block multipliers. *International Journal of Robust and Nonlinear Control*, 33(18), 11434–11457. doi:https://doi.org/10.1002/rnc.6952.

Morato, M.M., Normey-Rico, J.E., and Sename, O. (2020). Model predictive control design for linear parameter varying systems: A survey. *Annual Reviews in Control*, 49, 64–80. doi:https://doi.org/10.1016/j.arcontrol.2020.04.016.

Reiter, R., Nurkanović, A., Frey, J., and Diehl, M. (2023). Frenet-cartesian model representations for automotive obstacle avoidance within nonlinear mpc. *European Journal of Control*, 74, 100847. doi:https://doi.org/10.1016/j.ejcon.2023.100847. 2023 European Control Conference Special Issue.

Sename, O., Borrell, A.M., and Morato, M.M. (2024). On robust and predictive control approaches for linear parameter varying systems: Application to vehicle lateral control. *Foundations and Trends® in Systems and Control*, 11(4), 285–380. doi:10.1561/2600000032. URL `http://dx.doi.org/10.1561/2600000032`.

Shamma, J.S. (2012). *An Overview of LPV Systems*, 3–26. Springer US, Boston, MA. doi:https://doi.org/10.1007/978-1-4614-1833-7_1.