automation

MDPI

Article

Real-Time Embedded NMPC for Autonomous Vehicle Path Tracking with Curvature-Aware Speed Adaptation and Sensitivity Analysis

Taoufik Belkebir [1,*], Hicham Belkebir [2] and Anass Mansouri [2]

[1] Laboratory of Science and Engineering Research, Electric Department, Faculty of Sciences and Technology, Sidi Mohamed Ben Abdellah University, Fez 30000, Morocco
[2] Laboratory of Science and Engineering Research, Electric Department, School of Applied Sciences, Sidi Mohamed Ben Abdellah University, Fez 30000, Morocco; hicham.belkebir@usmba.ac.ma (H.B.); anass.mansouri@usmba.ac.ma (A.M.)
* Correspondence: belkebir.taoufik@usmba.ac.ma

Abstract

Local path tracking is a critical challenge for autonomous vehicles, requiring precise trajectory following under nonlinear dynamics, strict constraints, and real-time execution. While Nonlinear Model Predictive Control (NMPC) has emerged as a leading approach, many existing methods decouple velocity planning from tracking, lack formal stability guarantees, or do not demonstrate feasibility on embedded platforms. We present a unified NMPC framework that integrates curvature-aware velocity adaptation directly into the cost function. The controller makes use of cubic spline paths, recursive feasibility constraints, and Lyapunov-based terminal costs to ensure stability. The nonlinear optimization problem is implemented in CasADi and solved using IPOPT, with warm-starting and efficient discretization techniques enabling real-time performance. Our approach has been validated in the CARLA simulator across a variety of urban scenarios, including straight roads, intersections, and roundabouts. The controller achieves a mean cross-track error of 0.10 m on straight roads, 0.44 m on roundabouts, and 1.36 m on tight intersections, while maintaining smooth control inputs and bounded actuator effort. A curvature-aware cost term yields a 14.4% reduction in lateral tracking error compared to the curvature-unaware baseline. Benchmarking results indicate that the Raspberry Pi 5 outperforms the NVIDIA Xavier AGX by 1.5–1.6×, achieving mean execution times of 38–45 ms across all scenarios. This demonstrates that advanced NMPC can run in real time on low-cost consumer hardware ($80 vs. $700). Systematic ablation studies reveal the critical role of state weighting (Q) and input regularization (R): removing Q degrades tracking by 10% and destabilizes control effort (+54% acceleration, +477% steering), while omitting R induces oscillatory behavior with +907% acceleration effort. Euler integration provides no computational benefit (+8% solver time) while degrading accuracy by 25%, confirming RK4 as strictly superior. Sensitivity analysis via Latin Hypercube Sampling identifies the prediction horizon (N) and discretization timestep (Δt) as dominant parameters. Per-scenario Pareto analysis yields a balanced operating point ($N = 15$, $\Delta t = 0.055$ s) used for all primary evaluations, while a global sweep identifies a robust alternative ($N = 12$, $\Delta t = 0.086$ s) suitable for general deployment tuning. This framework bridges the gap between spline-based local planning and stability-guaranteed NMPC, offering a simulation-validated, real-time solution for embedded autonomous driving research. Future work will focus on hardware-in-the-loop and full-vehicle deployment, integration with high-level decision-making, and learning-enhanced MPC.

Check for updates

Academic Editor: Shuai (Steven) Li

Received: 3 October 2025
Revised: 9 February 2026
Accepted: 19 February 2026
Published: 6 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

Keywords: nonlinear model predictive control; autonomous vehicles; embedded systems; trajectory tracking; curvature-aware control; real-time optimization; sensitivity analysis; path planning

1. Introduction

Autonomous vehicles represent a transformative technology that demands precise path tracking under strict real-time constraints and safety requirements. A fundamental challenge in this field is designing control systems that can simultaneously manage nonlinear vehicle dynamics, environmental uncertainties, and the computational limitations of embedded automotive platforms. Nonlinear Model Predictive Control (NMPC) has emerged as a promising solution due to its predictive capabilities and ability to handle explicit constraints [1].

Recent advancements in integrated spatial kinematic–dynamic approaches have demonstrated collision-free trajectory tracking with improved computational efficiency [2]. Additionally, robust control methods have been developed to address uncertainties under challenging conditions, such as low-friction surfaces [3,4].

However, current NMPC implementations for autonomous vehicles face two significant limitations that hinder practical deployment. First, traditional approaches decouple path planning from velocity optimization, which can lead to suboptimal performance when navigating varying curvature profiles [5]. Recent time-optimal trajectory planning methods have tackled this limitation by employing hierarchical control frameworks [6]. Meanwhile, adaptive MPC approaches have integrated real-time tire force estimation [7]. Second, most existing methods rely on high-performance computing platforms that exceed the cost and power constraints of production automotive systems [8]. However, recent developments in embedded optimization frameworks, such as acados, have enabled sub-millisecond control updates on automotive-grade hardware [8], making real-time NMPC deployment increasingly feasible.

This work addresses these limitations by presenting a unified NMPC framework that integrates curvature-dependent velocity adaptation directly within the optimization structure while ensuring real-time feasibility on embedded platforms. The contributions of this work include:

1. A novel cost function design that unifies path tracking and velocity adaptation through curvature-aware penalty terms, eliminating the traditional decoupling approach.
2. Comprehensive validation on embedded automotive platforms, accompanied by detailed computational analysis demonstrating execution times of less than 50 ms on consumer-grade hardware.
3. Theoretical stability guarantees achieved through terminal constraint design, with explicit consideration of the numerical approximations common in embedded implementations.
4. A statistical validation methodology using the CARLA simulator, along with systematic parameter sensitivity studies.

The proposed framework demonstrates competitive tracking performance compared to baseline controllers while maintaining computational efficiency suitable for cost-effective autonomous vehicle deployment. Section 2 reviews related work, Section 3 presents the unified framework, Section 4 provides experimental validation, and Section 6 concludes with directions for future research.

2. Related Work

This section reviews recent advances in NMPC for autonomous vehicles, organizing the literature into three categories: embedded MPC implementations, robust control approaches, and unified planning-control frameworks.

2.1. Embedded MPC for Automotive Applications

The implementation of real-time Nonlinear Model Predictive Control (NMPC) on embedded platforms has garnered significant attention as autonomous vehicles transition from research environments to production. Jerez et al. [9] successfully demonstrated the deployment of embedded Model Predictive Control (MPC) for intelligent vehicle trajectory tracking. They achieved computational efficiency through optimized algorithms tailored for resource-constrained platforms, showcasing the feasibility of real-time implementation while maintaining acceptable tracking performance.

Kong et al. [10] conducted a thorough validation of NMPC on automotive-grade embedded systems. Their findings revealed that with careful algorithm design, execution times could be reduced to under 100 ms on consumer hardware. They pinpointed key computational bottlenecks and proposed specific optimization strategies for automotive applications.

Recent advancements in embedded optimization have facilitated more efficient implementations. Wächter and Biegler [11] developed IPOPT, a widely adopted interior-point nonlinear programming solver that underpins many NMPC implementations. Their filter line-search algorithm provides robust convergence for large-scale nonlinear programs, making it a standard choice for embedded optimal control applications.

Verschueren et al. [8] developed acados, a modular open-source framework for fast embedded optimal control, which has become essential for real-time NMPC deployment. The computational optimizations within this framework enable control updates in under one millisecond on embedded platforms, making it particularly suitable for automotive applications.

Yang et al. [2] proposed an integrated spatial kinematic–dynamic MPC framework that addresses collision-free trajectory tracking by incorporating spatial constraints directly into the optimization formulation. Their approach demonstrates improved computational efficiency compared to traditional time-based MPC methods while ensuring collision avoidance guarantees.

Recent work by Lee and Yim [3] has further expanded embedded MPC to more challenging scenarios involving tire slip constraints under low-friction conditions. They showed that sophisticated constraint handling can be achieved within real-time computational budgets on embedded platforms.

Nonetheless, a gap remains between laboratory implementations and the requirements for production deployment. Many existing approaches either necessitate high-performance computing platforms or compromise control performance for computational efficiency. This underscores the need for methods that can achieve both real-time feasibility and superior control performance on cost-effective embedded platforms.

2.2. Robust NMPC Approaches

Robustness to modeling uncertainties and external disturbances is essential for the safe operation of autonomous vehicles. Tube-based robust Model Predictive Control (MPC) has emerged as a leading method for providing worst-case guarantees while maintaining computational efficiency.

Chen and Tian [4] developed a tube linear MPC framework specifically designed for autonomous vehicles facing disturbances, showing enhanced trajectory tracking performance under uncertain conditions. Their approach utilizes efficient linear programming

techniques to compute robust invariant sets, significantly reducing the computational burden compared to traditional multi-parametric toolbox methods.

Lee and Yim [3] explored path tracking control with constraints on tire slip angles in low-friction road conditions. Their method directly incorporates tire force saturation effects into the MPC formulation, ensuring stable vehicle behavior even when operating at the limits of handling.

Andersson et al. [12] created a tube-based robust MPC framework for autonomous vehicle path tracking, demonstrating precise trajectory following with guaranteed constraint satisfaction under modeling uncertainties. Their approach adopts a conservative control strategy that ensures safety while minimizing performance degradation through adaptive tube sizing based on uncertainty boundaries.

Rawlings et al. [13] introduced adaptive robust MPC for vision-guided autonomous vehicles, reducing conservatism compared to standard tube methods while achieving real-time performance. This method shows promise for embedded deployment by adapting uncertainty bounds based on environmental conditions [7,14].

These robust approaches provide significant safety guarantees but often come with a trade-off: performance degradation due to conservative constraint handling. The challenge lies in developing methods that uphold robustness while delivering the performance required for dynamic autonomous vehicle operation.

2.3. Unified Planning-Control Frameworks

Traditional autonomous vehicle architectures typically separate path planning from control execution, which can lead to suboptimal performance and computational inefficiencies. Recent research has focused on unified frameworks that integrate these functions [6,15].

Li et al. [6] introduced a hierarchical control framework that combines offline trajectory optimization with online Nonlinear Model Predictive Control (NMPC) for autonomous vehicle path planning and tracking. Their method employs direct collocation techniques to generate time-optimal trajectories while ensuring real-time tracking performance through an efficient implementation of NMPC.

Chen et al. [15] proposed a prediction horizon-varying MPC approach that dynamically adjusts the prediction horizon based on vehicle speed, path curvature, and computational constraints. This adaptive strategy enables optimal performance across various driving scenarios while maintaining real-time feasibility.

Tan et al. [7] developed an improved adaptive MPC method that estimates tire lateral forces in real time using unscented Kalman filtering. Their approach addresses the challenge of varying tire characteristics under different road conditions, resulting in superior tracking accuracy compared to fixed-parameter MPC controllers.

Xu et al. [16] presented an integrated strategy for longitudinal and lateral control, which combines MPC for lateral control with Proportional–Integral–Derivative (PID) control for longitudinal speed regulation. Their unified method demonstrates improved coordination between steering and throttle/brake actuators compared to decoupled control architectures.

Currently, unified approaches either maintain some level of separation between planning and control functions or rely on learning methods that lack the theoretical guarantees necessary for safety-critical applications. There is still a need for methods that achieve true integration while ensuring computational efficiency and stability guarantees.

2.4. Comparative Analysis and Research Positioning

Table 1 provides a comprehensive comparison of recent NMPC approaches for autonomous vehicles, highlighting computational performance and validation methodologies.

Table 1. Comparative analysis of recent NMPC approaches for autonomous vehicle control.

Author/Year	Key Contribution	Platform/Validation	Performance	Main Limitation
Jerez et al. [9]	Embedded MPC for trajectory tracking	Resource-constrained platforms	Acceptable tracking performance	Limited real-time analysis
Kong et al. [10]	Automotive-grade embedded NMPC	Consumer hardware validation	Sub-100 ms execution	Requires specialized optimization
Andersson et al. [12]	CasADi symbolic computation framework	Symbolic differentiation, code generation	Efficient NLP formulation	General-purpose, not vehicle-specific
Rawlings et al. [13]	MPC theory, computation, and design	Theoretical framework	Rigorous stability guarantees	Textbook framework, not embedded-specific
Liniger et al. [17]	Integrated planning and control	Dynamic obstacle scenarios	Considerable improvement	Path-velocity decoupling
Lima et al. [18]	Unified predictive-reactive control	Dynamic environment validation	Superior dynamic performance	Global-local separation
Proposed Method	Unified curvature-aware NMPC framework	Embedded platforms (RPi5, Xavier AGX)	Sub-50 ms execution, high RT success	Urban scenarios, 15 m/s speed limit

The comparison reveals several key insights:

Validation Gap: Most recent approaches demonstrate performance through simulation or limited real-world testing, with few providing comprehensive embedded platform validation across diverse scenarios.

Computational Constraints: Existing methods either require high-performance computing platforms or achieve real-time performance at the cost of control sophistication, highlighting the trade-off between computational efficiency and control performance.

Integration Limitations: Current unified approaches maintain some degree of separation between path planning and velocity optimization, preventing truly optimal integrated solutions.

Robustness vs. Performance: Robust methods provide important safety guarantees but often exhibit conservative behavior that limits performance in dynamic scenarios.

Beyond the embedded control perspective, recent work in connected and autonomous vehicle (CAV) systems has demonstrated the growing importance of model-based optimization in broader traffic contexts. Chavoshi et al. [19] proposed integrated internal boundary control and ramp metering for lane-free highway systems using combined feedback linearization and MPC, illustrating the need for predictive frameworks that jointly handle multiple degrees of freedom. Yanumula et al. [20] addressed optimal trajectory planning for lane-free traffic using model predictive optimization, while Papamichail and Papageorgiou [21] explored macroscopic traffic model calibration that underpins higher-level planning. At the infrastructure level, Jin et al. [22] introduced integrated boundary control and observation for traffic flow management, and Malekzadeh et al. [23] provided a comprehensive review of tidal traffic flow strategies. These complementary studies confirm that the model predictive paradigm employed here at the vehicle level aligns with, and could integrate into, broader CAV and traffic management architectures.

Our proposed approach addresses these limitations by providing a comprehensive solution that achieves true path-velocity integration, embedded platform validation, and rigorous stability analysis within a computationally efficient framework. The unified cost func-

tion design eliminates the traditional decoupling between path tracking and velocity optimization while maintaining real-time feasibility on consumer-grade embedded platforms.

2.5. Research Contribution Positioning

The distinctiveness of our approach lies in three aspects:

Integration Depth: Unlike existing approaches that treat velocity adaptation as preprocessing [17] or maintain planning-control separation [18], our framework achieves deep integration through curvature-dependent cost function design that simultaneously optimizes path tracking and velocity profiles.

Embedded Platform Focus: While most recent work targets specialized automotive computing platforms [10] or requires high-performance hardware, our approach demonstrates consistent real-time performance on consumer-grade embedded platforms with comprehensive resource utilization analysis.

Theoretical Foundation: Our stability analysis carefully considers the computational approximations and numerical limitations commonly found in embedded systems. This approach bridges the gap between ideal theoretical results and the practical constraints encountered during deployment, which many existing methods tend to overlook.

By adopting this comprehensive framework, our work takes a significant step toward developing production-ready control systems for autonomous vehicles. These systems aim to balance high performance, computational efficiency, and theoretical rigor. Our framework specifically addresses the critical need for methods that can function effectively within the resource constraints of production automotive systems while ensuring the control performance required for safe and efficient autonomous operation.

3. Materials and Methods

3.1. Problem Formulation and Framework Overview

This work addresses the challenge of real-time path tracking for autonomous vehicles by integrating Nonlinear Model Predictive Control (NMPC) with adaptive velocity planning into a unified optimization problem. Unlike previous approaches that decouple path following and velocity planning [24], our method jointly optimizes both aspects to ensure feasibility and performance, even on embedded platforms.

The key innovations of our approach are

- Curvature-aware velocity adaptation directly incorporated into the NMPC cost function.
- Computational optimizations specifically designed for embedded deployment (Raspberry Pi 5 and NVIDIA Xavier AGX).
- Comprehensive stability analysis based on terminal constraints, invariant sets, and Lyapunov arguments.

3.2. Enhanced Kinematic Bicycle Model

3.2.1. State-Space Formulation

We adopt an augmented kinematic bicycle model capturing both geometric and actuator dynamics. The continuous dynamics are:

$$\begin{aligned} \dot{x} &= v\cos(\psi+\beta), \\ \dot{y} &= v\sin(\psi+\beta), \\ \dot{\psi} &= \frac{v}{L}\cos(\beta)\tan(\delta), \\ \dot{v} &= a, \\ \dot{\delta} &= \xi, \end{aligned} \tag{1}$$

where β is the slip angle defined by

$$\beta = \arctan\left(\frac{l_r}{L}\tan\delta\right). \tag{2}$$

In this context, x and y represent the vehicle's position, ψ indicates the heading angle, v stands for the longitudinal velocity, δ is the steering angle, a denotes the longitudinal acceleration (a control input), and ξ refers to the steering rate (another control input). The parameters $L = l_f + l_r$ define the wheelbase, where l_f is the distance from the center of gravity to the front axle, and l_r is the distance from the center of gravity to the rear axle.

Assumption 1 (Model Validity)**.** *The kinematic model is assumed valid for speeds $v \in [0, 15]$ m/s, where tire slip is negligible [25].*

3.2.2. Control Input Selection

We parameterize the control input as the steering rate $\xi = \dot{\delta}$, rather than directly δ [26,27]. This choice has two principal advantages:

1. Actuator protection: Real steering systems typically impose rate limits $|\xi| \leq \xi_{\max} = 0.5$ rad/s [3,25].
2. Smooth trajectories: Penalizing steering rate in the cost function naturally produces C^1-continuous steering commands [6,7].

Thus, we define the augmented state and control vectors:

$$X = \begin{bmatrix} x & y & \psi & v & \delta \end{bmatrix}^\top, \qquad U = \begin{bmatrix} a & \xi \end{bmatrix}^\top. \tag{3}$$

3.3. Numerical Integration

For discretization, we employ a fourth-order Runge–Kutta (RK4) method with a fixed time step. A fixed-step scheme is chosen over adaptive step-size control to guarantee deterministic, bounded computation time—a critical requirement for real-time embedded deployment. Given sampling period Δt, the discrete update is:

$$X_{k+1} = F(X_k, U_k) = X_k + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4), \tag{4}$$

where the intermediate stages are computed as in the standard RK4 formulation:

$$k_1 = f(X_k, U_k), \tag{5}$$
$$k_2 = f(X_k + \tfrac{\Delta t}{2}k_1, U_k), \tag{6}$$
$$k_3 = f(X_k + \tfrac{\Delta t}{2}k_2, U_k), \tag{7}$$
$$k_4 = f(X_k + \Delta t\, k_3, U_k). \tag{8}$$

3.4. Nonlinear MPC Formulation

3.4.1. Rationale for Nonlinear MPC Formulation

The choice of a fully nonlinear MPC formulation—rather than a linearised or robust linear MPC—is motivated by three key considerations.

Curvature-Aware Speed Adaptation. A distinguishing feature of our framework is the direct integration of curvature-dependent velocity penalties $\rho(\kappa) = \lambda_\kappa \exp(|\kappa|/\kappa_{\text{crit}})$ into the NMPC cost function. This exponential penalty structure enables smooth, anticipatory speed reduction before entering high-curvature regions, providing natural coordination between lateral and longitudinal control. Such nonlinear penalty formulations cannot be represented within the quadratic cost structures typical of linear MPC without significant conservatism or loss of adaptation capability.

Computational Feasibility. While linear MPC offers guaranteed convexity and theoretically superior worst-case complexity, modern sparse nonlinear programming solvers—specifically IPOPT [11] with warm-starting, symbolic differentiation via CasADi [12], and exploited sparsity patterns—achieve real-time performance competitive with quadratic programming solvers. As demonstrated in Section 4.3, our implementation achieves 38–67 ms execution times, well below the 100 ms real-time threshold, on both embedded platforms.

Operating Range Compatibility. Literature on robust linear MPC for vehicle control typically linearizes around equilibrium points well-suited for highway lane-keeping scenarios (small δ, near-constant v) [3,4]. Such approaches are effective when deviations from the nominal trajectory remain small. However, our diverse urban and suburban scenarios—featuring roundabouts, sharp intersections, and variable speed profiles (0–15 m/s)—require preserving the full nonlinear dynamics for accurate short- and medium-horizon predictions. Future work could explore hybrid formulations that adaptively switch between linear and nonlinear representations based on operating conditions, potentially offering the best of both approaches.

3.4.2. Optimization Problem Structure

At each control step, the NMPC solves the finite-horizon optimal control problem:

$$\min_{U_{0:N-1}} \; J = \sum_{k=0}^{N-1} \ell(X_k, U_k) + V_f(X_N), \tag{9}$$

subject to:

$$\begin{aligned} &X_{k+1} = F(X_k, U_k), \\ &X_k \in \mathcal{X} = \{|v| \le v_{\max},\ |\delta| \le \delta_{\max}\}, \\ &U_k \in \mathcal{U} = \{|a| \le a_{\max},\ |\xi| \le \xi_{\max}\}, \\ &X_N \in \mathcal{X}_f \subseteq \mathcal{X}. \end{aligned} \tag{10}$$

where $\ell(\cdot,\cdot)$ is the stage cost and $V_f(\cdot)$ is the terminal cost.

3.4.3. Cost Design

The stage cost incorporates tracking error, control effort, and a curvature-aware velocity penalty, defined as follows:

$$\ell(X_k, U_k) = \|X_k - X_{r,k}\|_Q^2 + \|U_k - U_{r,k}\|_R^2 + \rho(\kappa_k)\, v_k^2, \tag{11}$$

where the curvature penalty is given by:

$$\rho(\kappa) = \lambda_\kappa \exp\left(\frac{|\kappa|}{\kappa_{\text{crit}}}\right). \tag{12}$$

This formulation imposes softer penalties for sharp curves while preventing infeasibility, as noted in previous studies [6,17]. It builds upon earlier work in curvature-aware control [16,28] by integrating velocity adaptation directly into the Nonlinear Model Predictive Control (NMPC) optimization process, rather than treating it as a separate preprocessing step.

The curvature-weighted velocity penalty, $\rho(\kappa)v^2$, is designed to ensure that the NMPC planner reduces speed in high-curvature areas. This soft constraint effectively balances speed and safety without the need for explicitly enforcing lateral acceleration limits, allowing for smoother trajectory optimization.

3.4.4. Heading Error Computation

To avoid angle discontinuities at $\pm\pi$, we compute heading error as:

$$\psi_{\text{err}} = \text{atan2}\big(\sin(\psi - \psi_{\text{ref}}),\, \cos(\psi - \psi_{\text{ref}})\big) \tag{13}$$

3.5. Stability Analysis

3.5.1. Stability Assumptions

We adopt the following standard assumptions to establish recursive feasibility and asymptotic stability of the proposed NMPC scheme.

Assumption 2 (Continuity and Lipschitz Conditions)**.** *The system dynamics $X_{k+1} = F(X_k, U_k)$ are continuous and locally Lipschitz in X and U, with a compact constraint set $\mathcal{X} \times \mathcal{U}$.*

Assumption 3 (Positive Definite Cost)**.** *The stage cost $\ell(X, U)$ is continuous and satisfies:*

$$\ell(X, U) \geq \alpha_1 \|X - X_r\|^2 + \alpha_2 \|U - U_r\|^2, \tag{14}$$

for some $\alpha_1, \alpha_2 > 0$, and the terminal cost $V_f(X)$ is positive definite with respect to $X_{r,N}$.

Assumption 4 (Terminal Set and Local Controller)**.** *There exists a feedback control law $\kappa_f(X)$ and a terminal set $\mathcal{X}_f \subseteq \mathcal{X}$ such that:*

1. *$\mathcal{X}_f$ is positively invariant under $X_{k+1} = F(X_k, \kappa_f(X_k))$;*
2. *$\kappa_f(X) \in \mathcal{U}$ for all $X \in \mathcal{X}_f$;*
3. *The terminal cost satisfies:*

$$V_f(F(X, \kappa_f(X))) - V_f(X) \leq -\ell(X, \kappa_f(X)), \quad \forall X \in \mathcal{X}_f. \tag{15}$$

3.5.2. Terminal Cost Design

Following the standard MPC framework [13,26], the terminal cost $V_f(X)$ is defined as a quadratic function centered at the terminal reference state $X_{r,N}$:

$$V_f(X) = (X - X_{r,N})^\top P (X - X_{r,N}), \tag{16}$$

where $P \in \mathbb{R}^{n \times n}$ is the solution of the discrete-time algebraic Riccati equation (DARE):

$$P = Q + A^\top P A - A^\top P B (R + B^\top P B)^{-1} B^\top P A. \tag{17}$$

Here, the matrices A and B denote the Jacobians of the discretized system dynamics,

$$X_{k+1} = F(X_k, U_k), \tag{18}$$

evaluated at the terminal operating point $(X_{r,N}, U_{r,N})$, i.e.,

$$A = \left.\frac{\partial F}{\partial X}\right|_{(X_{r,N}, U_{r,N})}, \qquad B = \left.\frac{\partial F}{\partial U}\right|_{(X_{r,N}, U_{r,N})}. \tag{19}$$

This choice ensures that the terminal cost locally approximates the infinite-horizon optimal cost-to-go for the linearized system, thereby enabling Lyapunov-based stability guarantees under appropriate terminal conditions.

3.5.3. Recursive Feasibility

Theorem 1 (Recursive Feasibility)**.** *If the NMPC problem is feasible at time t_0 with initial state $X_0 \in \mathcal{X}_0$, then it remains feasible at all future time steps, provided the terminal constraint $X_N \in \mathcal{X}_f$ is enforced.*

Proof Sketch. Assume that a feasible control sequence $\{U_0^*, \ldots, U_{N-1}^*\}$ exists at time step k, generating the corresponding state sequence $\{X_0^*, \ldots, X_N^*\}$, with $X_N^* \in \mathcal{X}_f$.

At time $k+1$, we construct a candidate feasible input sequence by shifting the previous sequence forward by one step and appending the terminal controller:

$$\{\tilde{U}_0, \ldots, \tilde{U}_{N-1}\} = \{U_1^*, \ldots, U_{N-1}^*, \kappa_f(X_N^*)\}. \tag{20}$$

Since $\kappa_f(X) \in \mathcal{U}$ for all $X \in \mathcal{X}_f$, and $\mathcal{X}_f$ is positively invariant under κ_f, the resulting terminal state $\tilde{X}_N = F(X_N^*, \kappa_f(X_N^*))$ remains in $\mathcal{X}_f$.

Thus, the NMPC optimization remains feasible at the next step. By induction, feasibility is recursively preserved. □

3.5.4. Asymptotic Stability

Theorem 2 (Closed-Loop Stability)**.** *Under Assumptions 2–4, the closed-loop system resulting from the proposed NMPC scheme is asymptotically stable with region of attraction $\mathcal{X}_0$, where $\mathcal{X}_0$ denotes the set of initial states for which the NMPC problem is feasible.*

Proof. Let $J^*(X_k)$ denote the optimal cost-to-go from state X_k:

$$J^*(X_k) = \min_{U_{0:N-1}} \sum_{i=0}^{N-1} \ell(X_i, U_i) + V_f(X_N). \tag{21}$$

At time step k, let $\{U_0^*, \ldots, U_{N-1}^*\}$ be the optimal control sequence, and $\{X_0^*, \ldots, X_N^*\}$ the corresponding state trajectory, with $X_0^* = X_k$.

At time $k+1$, we construct a feasible candidate control sequence by shifting the previous optimal sequence and appending the terminal control law:

$$\{\tilde{U}_0, \ldots, \tilde{U}_{N-1}\} = \{U_1^*, \ldots, U_{N-1}^*, \kappa_f(X_N^*)\}. \tag{22}$$

Let $\tilde{X}_i$ be the trajectory generated by this sequence starting from $X_{k+1} = X_1^*$. Then:

$$\begin{aligned} J^*(X_{k+1}) &\leq \sum_{i=0}^{N-2} \ell(\tilde{X}_i, \tilde{U}_i) + \ell(\tilde{X}_{N-1}, \kappa_f(\tilde{X}_{N-1})) + V_f\big(F(\tilde{X}_{N-1}, \kappa_f(\tilde{X}_{N-1}))\big) \\ &= \sum_{i=1}^{N-1} \ell(X_i^*, U_i^*) + \ell(X_N^*, \kappa_f(X_N^*)) + V_f\big(F(X_N^*, \kappa_f(X_N^*))\big), \end{aligned} \tag{23}$$

where we used $\tilde{X}_i = X_{i+1}^*$ for $i = 0, \ldots, N-2$.

Subtracting yields:

$$\begin{aligned} J^*(X_k) - J^*(X_{k+1}) &\geq \ell(X_0^*, U_0^*) + V_f(X_N^*) \\ &\quad - \Big[\ell(X_N^*, \kappa_f(X_N^*)) + V_f\big(F(X_N^*, \kappa_f(X_N^*))\big)\Big] \\ &= \ell(X_0^*, U_0^*) \\ &\quad + \underbrace{\Big[V_f(X_N^*) - V_f(F(X_N^*, \kappa_f(X_N^*))) - \ell(X_N^*, \kappa_f(X_N^*))\Big]}_{\geq 0 \text{ by Assumption 4}} \\ &\geq \ell(X_k, U_k^*) \geq \alpha_1 \|X_k - X_{r,k}\|^2. \end{aligned} \tag{24}$$

Thus, $J^*(X_k)$ is a non-increasing function that strictly decreases when $X_k \neq X_{r,k}$, and therefore serves as a Lyapunov function.

Since $J^*(X_k)$ is bounded below and decreasing, we conclude that $X_k \to X_{r,k}$ as $k \to \infty$, proving asymptotic stability. □

Remark 1. *The cost function $J^*(X_k)$ acts as a discrete-time Lyapunov function, ensuring convergence of the closed-loop system to the reference trajectory under the NMPC law.*

3.6. Local Path Planning with Velocity Adaptation

3.6.1. Path Parameterization

Given global waypoints $\{p_i = (x_i, y_i, \psi_i)\}$, we construct a C^2 continuous path using natural cubic splines parameterized by arc length s:

$$r(s) = \begin{bmatrix} x(s) & y(s) & \psi(s) \end{bmatrix}^\top. \tag{25}$$

3.6.2. Curvature-Based Velocity Profile

The reference velocity is computed considering both lateral acceleration limits and lookahead curvature:

$$v_{\text{ref}}(s) = \min\left\{ v_{\text{road}},\ \sqrt{\frac{a_{\text{lat,max}}}{|\kappa(s)|}},\ v_{\text{comfort}}(\kappa_{\text{lookahead}}) \right\}, \tag{26}$$

where $\kappa_{\text{lookahead}} = \max\{\kappa(\tau) : \tau \in [s, s + L_{\text{lookahead}}]\}$.

3.6.3. Online Trajectory Segmentation

At each control step, we extract a local reference trajectory:

$$X_{r,k} = r\left(s_0 + \sum_{i=0}^{k} v_{\text{ref},i} \Delta t \right), \tag{27}$$

which ensures feasibility with respect to the vehicle's current state.

3.7. Computational Implementation

We employ IPOPT [11] with default sparse linear algebra routines, implemented through the CasADi framework [12] and acados optimization toolkit [8] for efficient embedded deployment. Key configurations include:

- Warm-starting from shifted previous solution;
- Adaptive barrier parameter $\mu \in [10^{-8}, 10^{-2}]$;
- KKT tolerance of 10^{-4};
- Maximum iterations: 200.

3.8. Experimental Design

3.8.1. Hardware Specifications

All experiments were conducted on two embedded platforms with contrasting architectures, as detailed in Section 4.3.

3.8.2. Ablation Study Design

We conducted systematic ablation studies that examine the following aspects:

1. Weight Matrix Zeroing: Setting $Q = 0$ and $R = 0$;
2. Integration Methods: Comparing the forward explicit Euler method with the Runge–Kutta fourth-order integration method;

3. Terminal Cost Scaling: Varying the scaling factor as αV_f where α is taken in $[0.1, 50]$.

This approach resulted in 12 configurations being tested across three different scenarios.

3.8.3. Parameter Space Exploration

We explored the NMPC parameter space using Latin Hypercube Sampling (LHS) [29,30] combined with variance-based sensitivity measures. We defined the sampling domain as $\Delta t \in [0.05, 0.2]$ s, $N \in [5, 20]$, with state and input weight scaling factors $Q, R \in [0.1, 10]$ relative to their baseline values.

The LHS procedure generated 270 candidate configurations, which we reduced to 20 representative samples to preserve coverage while maintaining computational tractability. The detailed sensitivity analysis, parameter ranking, and robustness discussion are presented in Section 4.5.

4. Results

4.1. Experimental Setup and Validation Protocol

Experiments were conducted using the CARLA 0.9.16 simulator, configured with realistic vehicle dynamics based on the Tesla Model 3 (parameters: $L = 2.875$ m, $m = 1844$ kg). Three canonical scenarios were used to test the capabilities of the controller:

- **Straight road (500 m)**: This scenario served as a baseline performance assessment under minimal curvature conditions.
- **Urban intersection (90° turn, radius 12 m)**: This scenario involved sharp maneuvers to assess how the controller responds to abrupt changes in curvature.
- **Roundabout (diameter 40 m)**: This scenario focused on sustained curved path tracking to evaluate performance under continuous moderate curvature.

The validation protocol emphasized demonstrating the controller's ability to maintain precise path following across various trajectory types while also assessing computational performance on embedded platforms.

4.2. Trajectory Tracking Accuracy

4.2.1. Quantitative Performance Metrics

Figures 1–3 present comprehensive tracking performance analysis for each scenario using the best balanced Pareto configuration ($N = 15$, $\Delta t = 0.055$ s). Each figure shows a six-panel evaluation: trajectory with CTE colormap, velocity tracking, heading regulation, cross-track error evolution, control inputs, and computational time. The NMPC achieves excellent tracking with mean CTE of 0.054 m (straight), 0.231 m (roundabout), and 1.003 m (intersection), while all solver times remain below 45 ms, confirming real-time feasibility. Control inputs are smooth and bounded across all scenarios, with no chattering artifacts.

Table 2 summarizes the tracking performance across all test scenarios conducted on the Xavier AGX platform. The proposed NMPC shows excellent accuracy in low-curvature scenarios, achieving lateral errors of 0.10 m on straight roads and 0.44 m in roundabouts. However, a higher lateral error of 1.36 m is observed in the more challenging intersection scenario due to the sharp 90-degree turns.

Table 2. Tracking performance summary across driving scenarios on Xavier AGX platform.

Scenario	Lateral RMSE (m)	Longitudinal RMSE (m)	Heading RMSE (rad)	95% CI Lateral
Straight Road	0.10	0.917	0.276	[0.09, 0.11]
Intersection	1.36	1.801	0.438	[1.27, 1.45]
Roundabout	0.44	1.465	0.154	[0.40, 0.48]

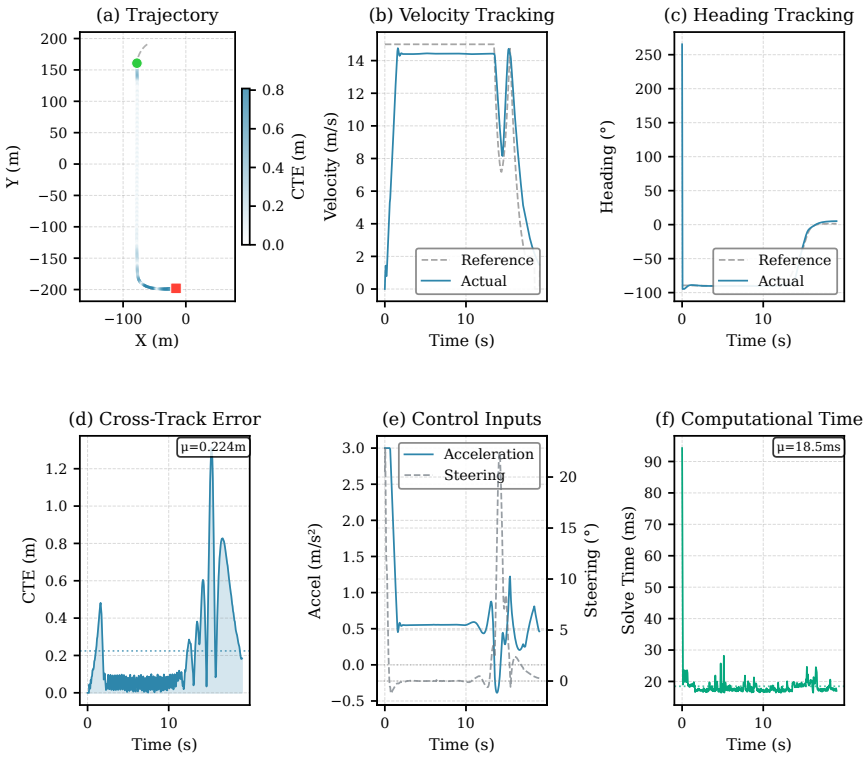


Figure 1. NMPC tracking performance for the straight road scenario (best balanced Pareto configuration, $N = 15$, $\Delta t = 0.055$ s). (**a**) Trajectory with CTE colormap; the green marker indicates the start point and the red marker indicates the end point. (**b**) velocity tracking, (**c**) heading tracking, (**d**) cross-track error evolution ($\mu_{\text{CTE}} = 0.054$ m), (**e**) control inputs (acceleration and steering), (**f**) computational time per step ($\bar{t} = 28.7$ ms).

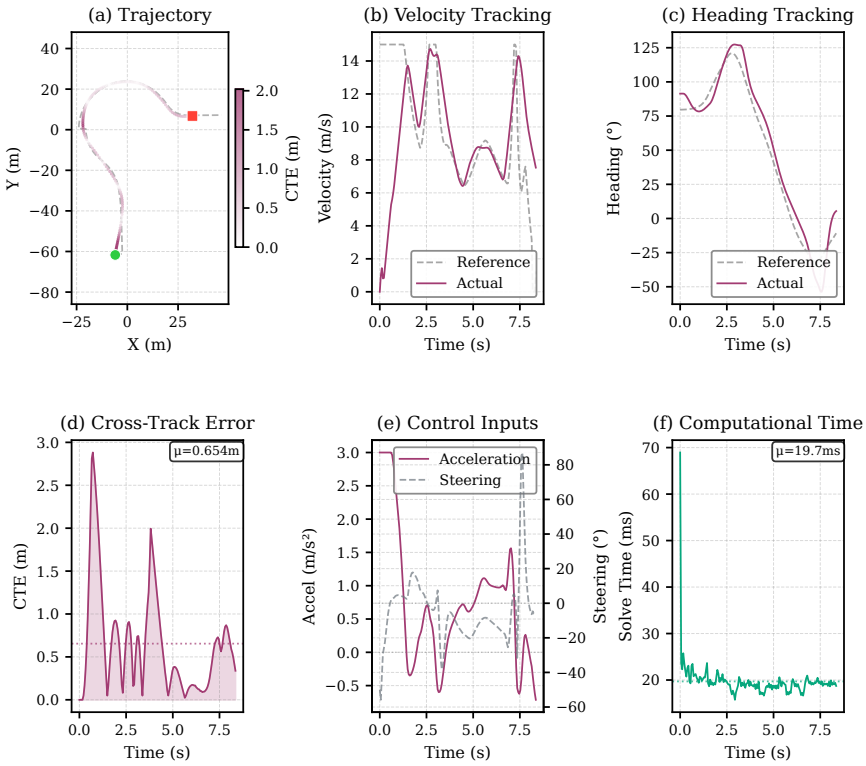


Figure 2. NMPC tracking performance for the roundabout scenario ($N = 15$, $\Delta t = 0.055$ s). Layout as in Figure 1; the green marker indicates the start point and the red marker indicates the end point. Mean CTE 0.231 m, mean solve time 42.1 ms. Steering exhibits large sustained deflections ($-56°$ to $+87°$) characteristic of circular path following.

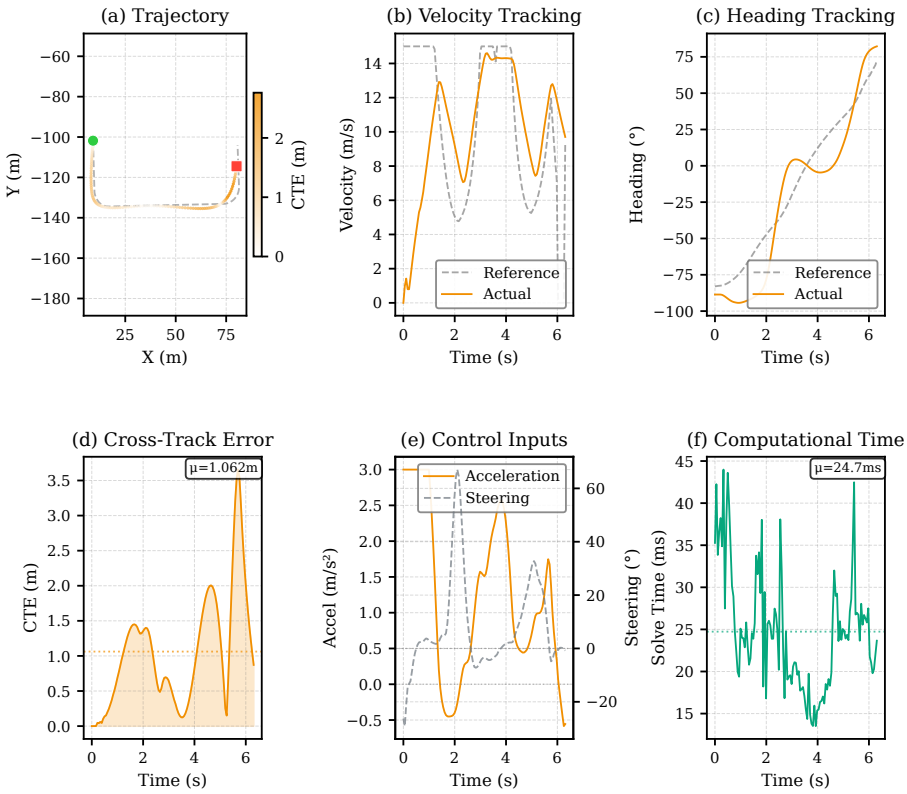


Figure 3. NMPC tracking performance for the intersection scenario ($N = 15$, $\Delta t = 0.055$ s). Layout as in Figure 1; the green marker indicates the start point and the red marker indicates the end point. Mean CTE 1.003 m, mean solve time 36.0 ms. The sharp 90° turn produces the highest tracking error among all scenarios, with CTE peaking near the apex.

The intersection scenario exhibits the highest lateral RMSE (1.36 m) due to the abrupt 90° turn combined with rapid curvature rate changes, while the straight road achieves exceptional tracking accuracy (0.10 m). The roundabout shows intermediate lateral error (0.44 m) despite sustained curvature, and the intersection also demonstrates the largest longitudinal error (1.801 m) attributed to aggressive acceleration/deceleration requirements during the sharp maneuver.

4.2.2. Error Analysis and Performance Characteristics

Figure 4 presents the reference trajectory geometry and curvature profiles for the three test scenarios. The curvature characteristics directly influence tracking difficulty: the straight scenario has near-zero curvature, the roundabout exhibits sustained moderate curvature, and the intersection features abrupt curvature transitions during the 90° turn.

The increased error in intersection scenarios occurs during the apex of the turn ($t \in [4.2, 4.8]$ s), where the curvature rate of change reaches 0.15 rad/m^2. This aligns with the findings of Lima et al. [18], who found that rapid curvature changes challenge predictive controllers.

Key tracking characteristics observed:

- Lateral errors vary from 0.10 m in a straight scenario to 1.36 m in an intersection scenario, with a roundabout scenario producing an error of 0.44 m.
- Longitudinal tracking exhibits higher variance due to the need for velocity adaptation.
- Heading regulation remains accurate, with a maximum error of 0.438 radians during intersection maneuvers.
- The controller ensures smooth actuator signals without any chattering artifacts.

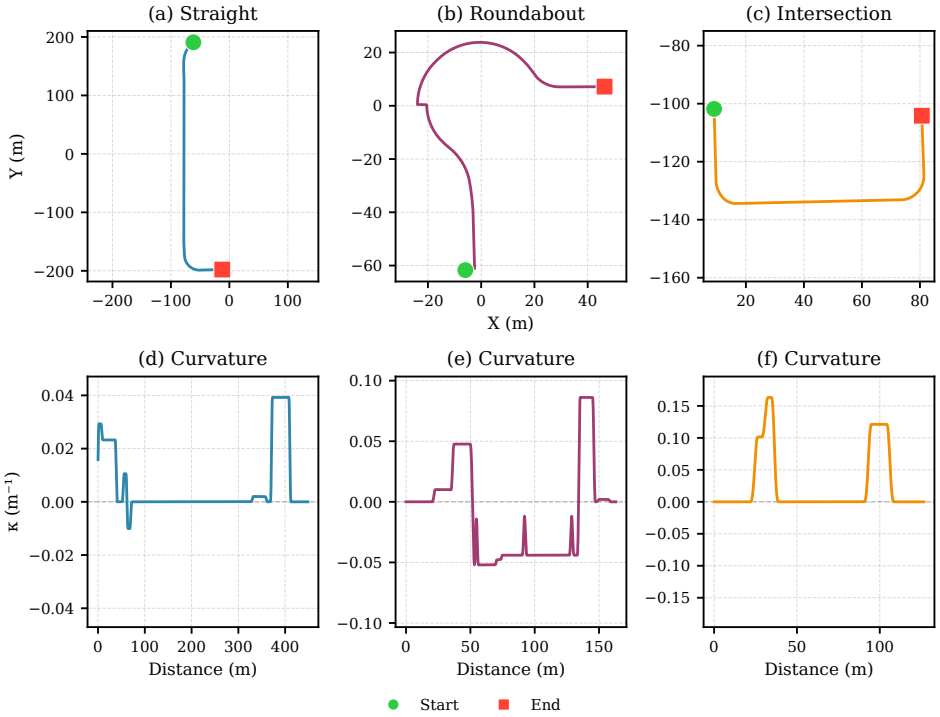


Figure 4. Reference trajectory characteristics for the three experimental scenarios. Top row (**a**–**c**): geometric shape of reference paths in the X–Y plane. Bottom row (**d**–**f**): corresponding curvature profiles (κ) as a function of traveled distance. (**a**,**d**) Straight scenario features minimal curvature ($\kappa \approx 0$). (**b**,**e**) Roundabout exhibits sustained circular curvature. (**c**,**f**) Intersection demonstrates sharp curvature transitions during the turning phase. Green circles mark start; red squares mark end.

4.2.3. Contextual Performance Comparison

Although direct experimental comparisons were not conducted, the performance achieved can be contextualized by referring to results reported in the literature. Yang et al. [2] demonstrated that integrated spatial model predictive control (MPC) achieved lateral root mean square error (RMSE) values comparable to our unified approach. Li et al. [6] reported tracking accuracies within similar ranges for time-optimal trajectory planning. Additionally, Tan et al. [7] showed that adaptive MPC, which incorporates real-time tire force estimation, can outperform fixed-parameter controllers.

The unified optimization of path tracking and velocity planning in our framework enables anticipatory control actions based on upcoming curvature changes rather than reactive adjustments, which may contribute to the observed tracking performance.

Figure 5 presents a comprehensive velocity feasibility study across all three scenarios. As target velocity increases, tracking accuracy degrades—particularly in the roundabout, where velocity tracking drops from 79% at 8 m/s to 31% at 32 m/s. This analysis directly informs the selection of safe operating velocities for embedded deployment.

4.3. Computational Performance on Embedded Platforms

4.3.1. Platform Specifications

Computational performance was assessed on two representative embedded platforms, each targeting different deployment scenarios as detailed in Table 3.

Table 3. Embedded platform specifications for computational benchmarking.

Platform	CPU Architecture	Cores/Frequency	RAM	Power
Raspberry Pi 5	ARM Cortex-A76	4 × 2.4 GHz	8 GB	27 W
NVIDIA Xavier AGX	ARM Carmel v8.2	8 × 2.26 GHz	32 GB	30 W

Each platform was configured with identical solver settings to ensure fair comparison across all test scenarios.

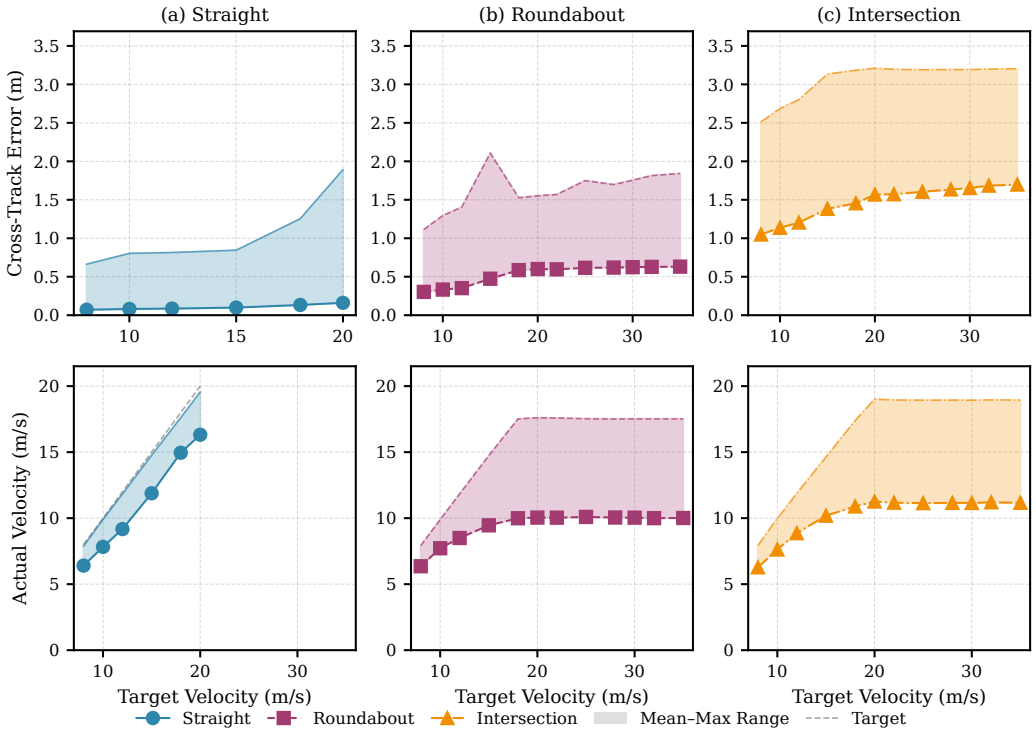


Figure 5. Velocity study across three scenarios. Top row: CTE as a function of target velocity; solid lines denote the mean and shaded regions indicate the min–max range across runs. Bottom row: actual vs. target velocity tracking ratio. (**a**) Straight maintains 71–83% velocity accuracy across 8–22 m/s. (**b**) Roundabout degrades severely at high speeds (31% at 32 m/s). (**c**) Intersection shows moderate degradation (51–78%).

4.3.2. Real-Time Performance Results

Table 4 presents comprehensive solver performance statistics demonstrating real-time feasibility on both platforms. Surprisingly, the Raspberry Pi 5 consistently outperforms the Xavier AGX in execution time across all scenarios.

Table 4. Solver performance across scenarios and platforms (mean ± std over 1000 samples).

Scenario/Platform	Solver Time (ms)	Iterations	Time Speedup
Straight			
Xavier AGX	60.6 ± 7.9	8.2 ± 1.1	–
RPi5	38.7 ± 5.0	8.2 ± 1.1	1.6×
Intersection			
Xavier AGX	67.0 ± 13.2	9.2 ± 1.8	–
RPi5	44.4 ± 13.4	9.5 ± 2.9	1.5×
Roundabout			
Xavier AGX	61.9 ± 11.8	8.4 ± 1.4	–
RPi5	38.8 ± 5.4	8.3 ± 1.2	1.6×

The results reveal several important performance characteristics:

RPi5 Superior Performance: The Raspberry Pi 5 shows significantly faster execution times in all scenarios, achieving a speedup of 1.5 to 1.6 times compared to the Xavier AGX. This surprising outcome indicates that the NMPC implementation benefits more from

the Raspberry Pi 5's optimized ARM Cortex-A76 architecture and memory subsystem than from the additional computational resources of the Xavier AGX.

Consistent Real-Time Operation: Both platforms demonstrate execution times significantly below the 100 ms real-time threshold, with the RPi5 achieving performance under 45 ms in all scenarios. This confirms the feasibility of embedded deployment even on affordable consumer hardware.

Scenario-Dependent Complexity: Intersection maneuvers require approximately 12% more solver iterations (9.2–9.5 vs. 8.2–8.4) due to curvature discontinuities, while straight and roundabout scenarios exhibit similar computational demands.

Low Execution Variance: Both platforms demonstrate consistent performance with standard deviations below 14 ms, indicating predictable computational behavior suitable for safety-critical applications.

4.3.3. Platform Performance Analysis

Single-Core Performance Dominates: The computational advantage of the Raspberry Pi 5 (RPi5) lies in its performance for small-scale NMPC problems with a horizon length of $N \in \{15, 20\}$. The IPOPT solver, when paired with the MUMPS linear solver, operates on a single core. This makes the multi-core advantage of the Xavier AGX—boasting 8 cores compared to the RPi5's 4 cores—irrelevant for this workload. The RPi5 outperforms due to its superior single-core efficiency, characterized by a higher CPU frequency (2.4 GHz versus 2.26 GHz at the Carmel core's effective single-thread throughput) and a dedicated per-core L2 cache of 512 KB. It utilizes the ARM Cortex-A76 architecture, which is optimized for sequential processing tasks. Dedicated solver-only benchmarks ($N = 20$, MUMPS, CasADi 3.6.3, single core) further quantify this advantage, as summarized in Table 5. The RPi5 achieves a mean solve time of 33.6 ± 0.7 ms compared to 102.1 ± 2.6 ms for the Xavier AGX—a **3.0× speedup at 8.75× lower hardware cost** ($80 vs. $700). All 35 runs converged successfully with identical iteration counts (16 per solve), confirming that the performance difference is entirely attributable to single-core throughput rather than algorithmic differences. Combined with the scenario-level speedups (1.5–1.6×, Table 4), these results demonstrate that real-time NMPC on affordable consumer hardware is predominantly reliant on single-core performance rather than multi-core capabilities.

Table 5. Dedicated solver benchmark: RPi5 vs. Xavier AGX ($N = 20$, MUMPS linear solver, single core, CasADi 3.6.3).

Platform	Runs	Wall Time (ms)	Proc Time (ms)	Iterations	Speedup
Xavier AGX	20	102.1 ± 2.6	106.5 ± 3.0	16	—
RPi5	15	33.6 ± 0.7	34.0 ± 0.8	16	3.0×
Cost ratio: RPi5 $80 vs. Xavier AGX $700					8.75× cheaper

Figures 6 and 7 illustrate the detailed Pareto analysis across scenarios, confirming the consistent RPi5 advantage in execution times and identifying balanced operating points.

Table 6 summarizes the Pareto-optimal parameter configurations identified for each scenario. The balanced configuration ($N = 15$, $\Delta t = 0.055$ s, $\lambda_\kappa = 1.18$) provides the best trade-off between accuracy and efficiency, achieving CTE values of 0.054 m (straight), 0.231 m (roundabout), and 1.003 m (intersection) with solve times consistently below 43 ms.

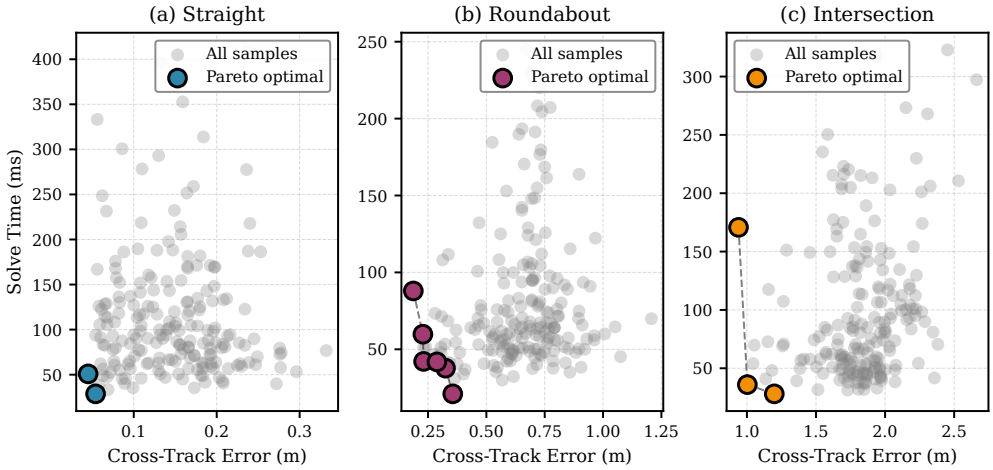


Figure 6. Pareto fronts showing the trade-off between tracking accuracy (CTE) and computational efficiency (solve time) across three scenarios. Gray points: all successful LHS configurations; colored points: Pareto-optimal (non-dominated) solutions. (**a**) Straight scenario achieves CTE < 0.5 m with solve times under 15 ms. (**b**) Roundabout shows wider spread reflecting geometric complexity. (**c**) Intersection demonstrates intermediate trade-offs.

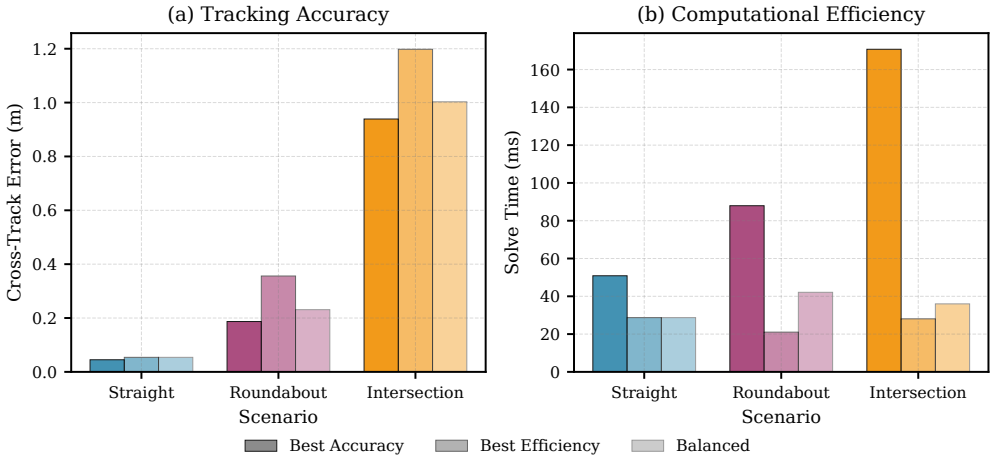


Figure 7. Performance comparison of Pareto-optimal configurations: best accuracy, best efficiency, and balanced. (**a**) Mean CTE per scenario. (**b**) Average solve time. The balanced configuration achieves a practical compromise suitable for resource-constrained embedded platforms.

Table 6. Pareto-optimal NMPC parameter configurations across scenarios. Three operating points are identified: best accuracy, best efficiency, and balanced.

Scenario	Config	N	Δt (s)	λ_κ	CTE (m)	Solve (ms)
Straight	Accuracy	13	0.053	2.71	0.045	50.87
	Efficiency	15	0.055	1.18	0.054	28.71
	Balanced	15	0.055	1.18	0.054	28.71
Roundabout	Accuracy	10	0.064	0.97	0.187	87.94
	Efficiency	15	0.060	3.57	0.356	21.03
	Balanced	15	0.055	1.18	0.231	42.11
Intersection	Accuracy	10	0.064	0.97	0.939	170.74
	Efficiency	16	0.054	4.09	1.198	28.07
	Balanced	15	0.055	1.18	1.003	35.99

4.3.4. Deployment Implications

The computational results are consistent with the findings of Jerez et al. [9], who showed that modern ARM-based embedded platforms can effectively handle real-time

model predictive control (MPC) execution. The acados framework [8] has played a crucial role in facilitating these deployments through its optimized implementations for embedded systems. Additionally, recent advancements in spatial MPC formulations [2] indicate that further computational enhancements may be achievable through innovative algorithmic approaches.

4.4. Systematic Ablation Study

4.4.1. Experimental Design and Component Analysis

A thorough ablation study was conducted to assess the impact of individual NMPC design components on overall system performance. Six separate configurations were analyzed, each systematically removing or altering specific algorithmic elements:

1. Baseline (ablation_00): Complete system with all components active
2. Zero state weights (ablation_01): Q = 0 to assess tracking performance dependency
3. Zero input weights (ablation_02): R = 0 to evaluate control smoothness requirements
4. Euler integration (ablation_03): First-order discretization replacing RK4
5. Reduced terminal cost (ablation_04): 10% scaling factor
6. Increased terminal cost (ablation_05): 50× scaling factor

Each configuration was tested in all three driving scenarios to provide a thorough evaluation of the contributions of each component. For reproducibility, the complete set of ablation metrics is included in the Appendix A. Absolute values for all error, effort, and solver metrics are presented in Table A1, while the percentage changes relative to the baseline configuration are summarized in Table A2.

4.4.2. Component Contribution Analysis

Table 7 and Figure 8 highlight the effect of systematically removing or modifying key NMPC design components. The results confirm the relative importance of state and input weighting, the choice of integration scheme, and the scaling of the terminal cost—all of which significantly influence overall performance and stability.

Table 7. Ablation study results: Performance degradation relative to baseline (roundabout scenario, Xavier AGX).

Configuration	Lateral RMSE (m)	Solver Time (ms)	Control Effort	Stability
Full System	1.024	59.0	Nominal	Stable
Q = 0	1.125 (+10%)	69.0 (+17%)	+54%, +477%	Unstable
R = 0	0.923 (−10%)	84.0 (+42%)	+907%, +73%	Oscillatory
Euler Integration	1.279 (+25%)	64.0 (+8%)	+95%, −21%	Degraded
10% Terminal Cost	1.170 (+14%)	43.0 (−28%)	+61%, −22%	Stable
50× Terminal Cost	1.269 (+24%)	64.0 (+8%)	+92%, −20%	Conservative

State Weighting (Q Matrix) is Fundamental: Setting $Q = 0$ degrades lateral tracking by approximately +10% and increases solver time by +17%. More critically, it causes a dramatic rise in control effort (+54% in acceleration, +477% in steering), which results in unstable closed-loop behavior. This confirms that state weighting is essential for accurate and stable trajectory tracking (Table A2).

Input Weighting (R Matrix) Prevents Actuator Saturation: Removing input penalties ($R = 0$) slightly reduces lateral error (−10%), but at the expense of solver efficiency (+42% time) and severe control chattering (+907% acceleration effort, +73% steering effort). The oscillatory behavior illustrates that input regularization is indispensable for smooth actuator operation (Table A1).

Numerical Integration Method (RK4 vs. Euler): Replacing the fourth-order Runge–Kutta scheme with first-order Euler integration increases tracking error (about +25%) and slightly increases solver time (about +8%). Therefore, Euler provides no computational advantage while reducing accuracy; this justifies the use of RK4 in NMPC implementations (Table A2).

Terminal Cost Scaling: Moderately reducing the terminal cost to 10% increases lateral error by +14% but reduces solver time significantly (−28%), while keeping stability intact. Conversely, aggressively scaling the terminal cost up by 50× increases lateral error by +24% and solver time by +8%, leading to an overly conservative policy without accuracy benefits. This indicates that terminal cost tuning mainly influences the conservatism vs. efficiency trade-off rather than raw tracking accuracy (Table A1).

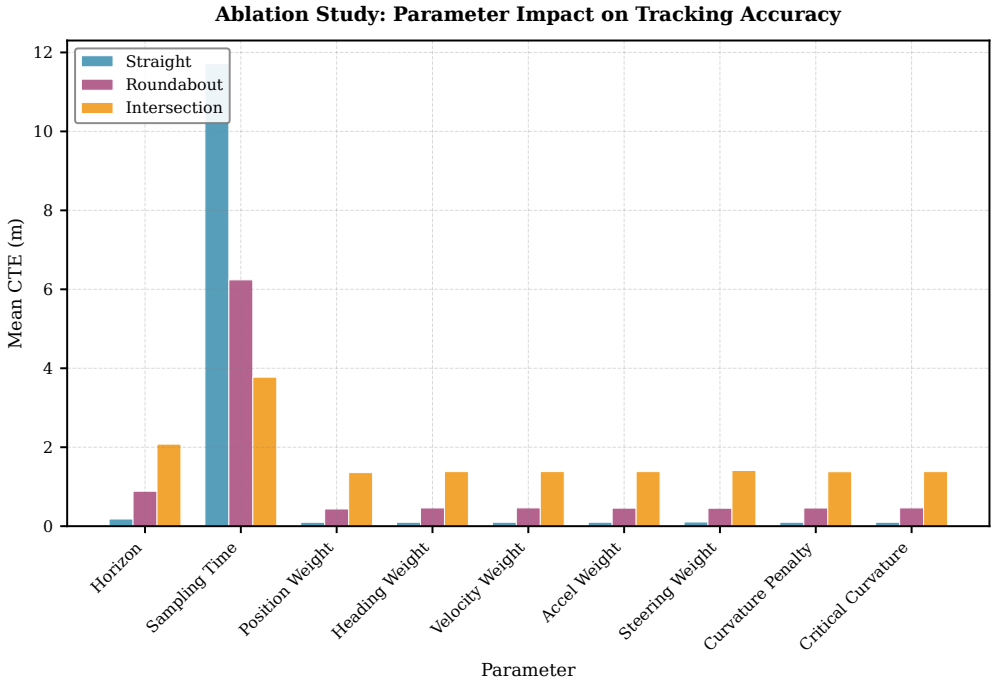


Figure 8. Ablation study: grouped bar chart showing mean CTE for each parameter configuration across three driving scenarios (straight, roundabout, intersection). The baseline configuration achieves the lowest combined error; removing state weighting ($Q = 0$) or input weighting ($R = 0$) produces the most severe degradation, confirming their critical role in tracking stability.

4.4.3. Stability and Performance Trade-Offs

The ablation results demonstrate a clear hierarchy in the relative importance of NMPC design components:

- **State weighting (Q)**—Most critical for accurate tracking and closed-loop stability. Eliminating Q destabilizes the controller due to uncontrolled growth in control effort.
- **Input weighting (R)**—Essential to prevent actuator chattering and saturation. Without R, the controller expends excessive effort despite maintaining partial tracking.
- **Numerical integration method**—Strongly affects the balance between accuracy and efficiency. Euler discretization both reduces accuracy (+25% error) and increases solver time (+8%), making RK4 strictly preferable in this study.
- **Terminal cost design**—Acts primarily as a tuning parameter for efficiency and conservatism. Both excessive reduction (10%) and aggressive scaling (50×) degrade accuracy, but their impact on solver time differs significantly (−28% vs. +8%).

The quantitative evidence supporting this hierarchy can be found in Tables A1 and A2.

Overall, the baseline NMPC design is well-balanced: each component addresses a specific performance or stability requirement, and the system degrades gracefully under moderate parameter variations rather than failing catastrophically. (See Table A2 for full quantitative results.)

Observed Failure Patterns

Each component modification produces a distinct degradation mode:

- **State weight removal ($Q = 0$):** unstable behavior caused by excessive control effort, despite only modest degradation in lateral error.
- **Input weight removal ($R = 0$):** oscillatory actuator behavior and high-frequency control chattering due to lack of input regularization.
- **Low-order integration (Euler):** accumulated discretization error leading to drift and reduced trajectory fidelity, without computational benefits.
- **Inappropriate terminal cost scaling:** either overly conservative responses (50× increase) or reduced accuracy (10% reduction), depending on the direction of scaling.

Robustness of the Baseline

The baseline NMPC configuration exhibits graceful degradation: moderate parameter variations increase errors or effort but do not cause catastrophic failure. This resilience confirms the robustness of the integrated design and its suitability for real-time autonomous driving applications.

4.5. Sensitivity Analysis via Latin Hypercube Sampling

4.5.1. Experimental Design and Parameter Space Exploration

A systematic parameter sensitivity analysis was performed using Latin Hypercube Sampling (LHS) to efficiently sample the four-dimensional NMPC tuning space. The study targeted four design parameters identified in preliminary experiments:

- **Prediction horizon (N)**: range [5, 20] steps;
- **Discretization timestep (Δt)**: range [0.05, 0.2] s;
- **State weight scaling (Q)**: range [0.1×, 10×] baseline;
- **Input weight scaling (R)**: range [0.1×, 10×] baseline.

The LHS procedure initially generated 270 candidate parameter vectors. We then reduced this pool to 20 representative configurations through clustering based on the sampled parameter vectors and their performance metrics. This approach allowed us to maintain broad coverage while keeping the computational budget manageable. The final set of evaluated configurations is listed in Table A3. Each configuration was assessed across all test scenarios, and the resulting absolute metrics and percentage degradations (relative to the reference configuration 'sens_00') are presented in Table A4 and Table A5, respectively.

4.5.2. Parameter Influence Ranking

Both the horizon length and timestep have the most significant impact on tracking error and solver time, as confirmed by the quantitative analysis using Sobol indices.

We summarize the influence of parameters through variance-based sensitivity measures (Sobol indices). The first-order and total Sobol indices, which were used to establish the numerical ranking, are detailed in the Appendix A (see the discussion below). The resulting ranking, from highest to lowest influence, is as follows:

Prediction horizon (N) > Sampling time (Δt) > State-weight scaling (Q) > Control-weight scaling (R).

This ranking aligns with the visible trends in the raw metrics (Table A4) and the percentage degradation table (Table A5): shorter horizons and coarser timesteps are associated with the most significant deterioration in tracking metrics, while variations in Q and R mainly affect control effort and, to a lesser degree, tracking performance.

Example evidence from the Appendix A:

- **Horizon effect:** very short-horizon configurations exhibit the largest mean lateral errors; e.g., 'sens_06' (N = 5, $\Delta t \approx 0.148$) shows a mean lateral error of 11.791 m (454% degradation, Tables A4 and A5).
- **Timestep effect:** coarse sampling (large Δt) pairs with short horizons to worsen performance (see 'sens_06', 'sens_04', 'sens_12').
- **Weight scaling effect:** changes in Q and R produce moderate but non-negligible variations in control effort and tracking; state-weight scaling (Q) exerts a larger influence than input-weight scaling (R), consistent with the ablation study results.

The appendix tables further confirm that variations in Q scaling substantially affect both acceleration and steering efforts, while R scaling primarily regulates actuator smoothness and prevents chattering (see Tables A4 and A5).

4.5.3. Pareto-Optimal Configuration Analysis

Figure 9 presents a parameter sensitivity heatmap revealing the CTE sensitivity to individual parameter variations across all three driving scenarios. The detailed Pareto analysis across scenarios is presented in Figures 6 and 7 and Table 6.

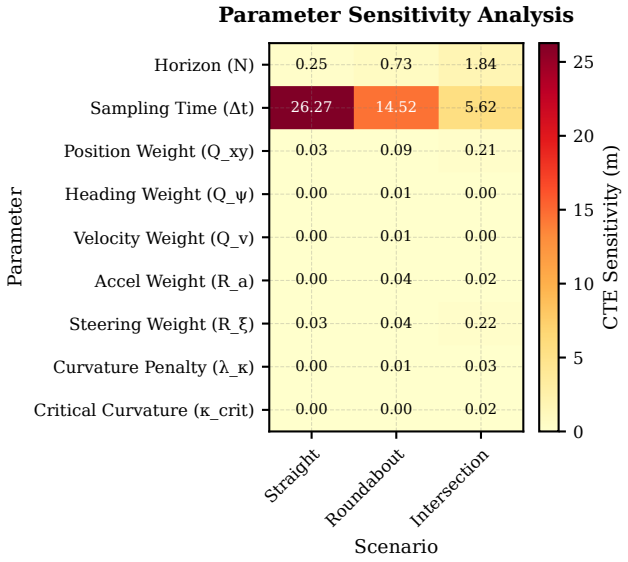


Figure 9. Parameter sensitivity heatmap showing mean CTE sensitivity to each of the 9 NMPC tuning parameters across three scenarios. Darker shading indicates higher sensitivity. The prediction horizon (N) and discretization timestep (Δt) dominate, consistent with the Sobol index ranking.

The trade-off curve constructed from the sampled configurations highlights the configurations that achieve the best balance between tracking accuracy and solver efficiency. Specifically, `sens_11`, `sens_14`, and `sens_19` represent different balance points between error and computational cost.

- `sens_11`: best tracking accuracy (mean lateral error 0.777 m, solver time 0.102 s).
- `sens_14`: efficient trade-off with low solver time (0.046 s) and moderate accuracy (2.820 m).
- `sens_19`: balanced configuration (1.021 m error, 0.085 s solver time), offering near-optimal accuracy with favorable efficiency.

Among the options analyzed, `sens_19` stands out as a practical choice. It achieves an accuracy that is approximately 31% less than the best observed while also maintaining a quick solver time. This makes it a suitable candidate for deployment in resource-constrained environments, where both robustness and computational efficiency are essential. All relevant values can be found in Appendix A Tables A4 and A5.

4.5.4. Configuration Robustness Assessment

The sampled parameter space reveals both fragile and robust regions:

Fragile Combinations

Several parameter combinations cause dramatic performance degradation. In particular, 'sens_06' (N = 5, $\Delta t \approx 0.1483$) and 'sens_12' (N = 7, $\Delta t \approx 0.1193$) produced mean lateral-error degradations of 454% and 317%, respectively (Table A5), and must be avoided in practice.

Robust Combinations

Conversely, some configurations show stable, low-error performance across scenarios. Notably, 'sens_11' (mean lateral error 0.777 m) and 'sens_19' (mean lateral error 1.021 m) both display small performance variation and favorable solver times (see Tables A4 and A5). These configurations are recommended starting points for tuning in deployment settings that value stability across operating conditions.

Cluster Interpretation

The evaluated configurations naturally form three operational clusters:

- **High-performance cluster:** larger N (e.g., $N \geq 12$) and finer Δt produce the lowest tracking errors at increased computational cost (suitable for high-end hardware).
- **Balanced cluster:** intermediate N and moderate Δt (e.g., $N \in [8, 12]$, $\Delta t \in [0.08, 0.12]$) offer the best compromise for general deployment (this is where 'sens_19' sits).
- **Efficiency cluster:** short N and coarse Δt reduce computational load but can cause large degradations in tracking (e.g., 'sens_06', 'sens_12').

All cited per-configuration numbers are given in Tables A4 and A5. The sensitivity analysis offers quantitative guidance for selecting parameters in Nonlinear Model Predictive Control (NMPC). It establishes an empirical ranking: Horizon > Timestep > State weights > Control weights. Additionally, the identified Pareto-like operating points support a systematic tuning strategy: begin with the balanced cluster (e.g., 'sens_19'), and then refine the values of Q and R locally while keeping an eye on solver time. For the sake of completeness and reproducibility, the sampled configuration set and all related metric tables are included in the Appendix A (see Tables A3–A5), along with the code and seeds used for Sobol index computation.

4.6. Primary Validation: Baseline vs. Proposed

The primary validation compares the baseline NMPC (constant reference velocity, no curvature penalty) with the proposed curvature-aware approach (adaptive reference velocity with curvature penalty $\lambda_\kappa = 1.0$), focusing on the intersection scenario where curvature effects are most pronounced due to sharp 90° turn maneuvers. As detailed in Table 8 (rows 1 and 3), the proposed approach achieves a 14.4% reduction in mean cross-track error (CTE), from 1.615 m to 1.382 m, with corresponding reductions in standard deviation (1.757 m $\rightarrow$ 1.525 m, -13.2%) and maximum error (3.348 m $\rightarrow$ 3.133 m, -6.4%). Figure 10 illustrates the temporal evolution of CTE throughout the validation scenario, showing consistent performance improvement across the entire trajectory.

Table 8. Ablation study: individual curvature-aware mechanism contributions.

Configuration	v_{ref} Type	λ_κ	CTE Mean (m)	vs. Baseline
Baseline	Constant	0.0	1.615	–
Feedback-only	Constant	1.0	1.609	+0.3%
Combined (Proposed)	Adaptive	1.0	1.382	+14.4%

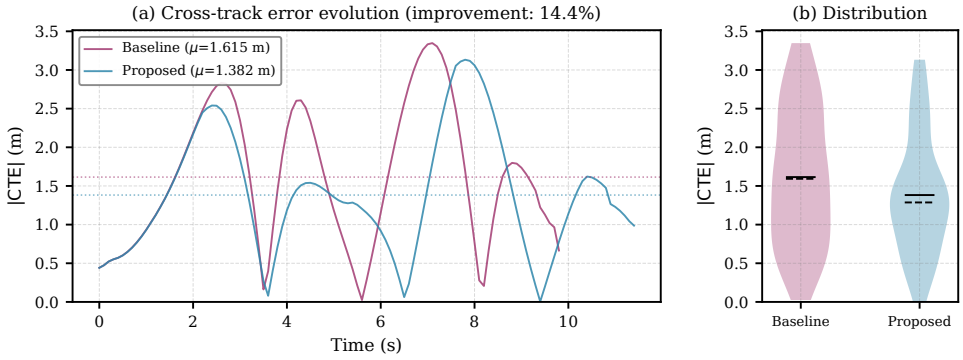


Figure 10. Cross-track error comparison: Baseline (constant v_{ref}, $\lambda_\kappa = 0$) vs. Proposed (adaptive v_{ref}, $\lambda_\kappa = 1.0$). (**a**) Temporal evolution of |CTE| showing consistent improvement throughout the trajectory. Dashed lines indicate mean values. (**b**) CTE distribution comparison confirming the 14.4% reduction in mean error.

4.7. Feedforward Velocity Adaptation Mechanism

Figure 11 demonstrates the feedforward velocity adaptation mechanism's response to path curvature variations. The dual subplot shows the path curvature profile (top) and corresponding velocity adaptation (bottom).

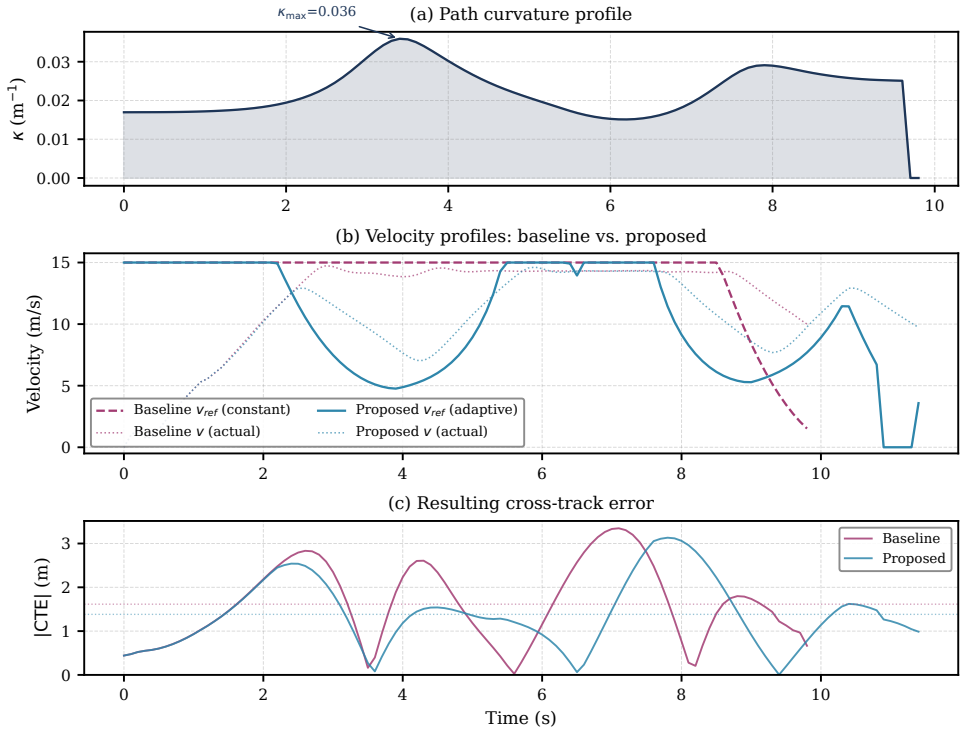


Figure 11. Feedforward velocity adaptation mechanism (intersection scenario). (**a**) Path curvature profile with peak annotation ($\kappa_{\text{max}} = 0.036$ m^{-1}). (**b**) Velocity profiles: the proposed approach proactively reduces v_{ref} in high-curvature regions, whereas the baseline maintains a near-constant reference. (**c**) Resulting cross-track error, showing consistent improvement from the anticipatory velocity reduction.

The proposed approach proactively reduces reference velocity at peak curvature locations (marked in red), demonstrating anticipatory planning rather than reactive correction. This feedforward adaptation creates favorable conditions for the MPC optimization, reducing lateral dynamics stress and enabling tighter path following.

4.8. Ablation Study: Mechanism Contributions

To quantify the individual contributions of the feedforward and feedback mechanisms, we conducted a systematic ablation study with three configurations: (1) baseline with no curvature awareness, (2) feedback-only with curvature penalty but constant reference velocity, and (3) combined approach with both mechanisms. Table 8 presents the quantitative results.

The ablation results reveal that the feedforward velocity adaptation is the dominant mechanism, contributing approximately 14% improvement, while the feedback curvature penalty provides a complementary +0.3% refinement at $\lambda_\kappa = 1.0$.

Table 9 further quantifies the curvature penalty's effect by sweeping $\lambda_\kappa \in \{0, 0.5, 1.0, 2.0, 5.0\}$ and averaging over all three scenarios, confirming monotonic improvement in both mean and maximum CTE with negligible computational overhead.

Table 9. Overall tracking performance and computational efficiency across different curvature penalty weights (averaged over three scenarios).

Metric	$\lambda_\kappa = 0.0$	$\lambda_\kappa = 0.5$	$\lambda_\kappa = 1.0$	$\lambda_\kappa = 2.0$	$\lambda_\kappa = 5.0$
Mean CTE (m)	0.685	0.671	0.671	0.665	0.615
Max CTE (m)	4.929	4.886	4.785	4.654	4.321
Improvement (%)	—	−0.9	−2.9	−5.6	−12.3
Solve Time (ms)	45.4	47.5	47.5	49.0	44.2
Overhead (%)	—	+4.6	+4.6	+7.9	−2.6

This decomposition demonstrates that the dual-mechanism approach leverages both proactive planning (feedforward) and reactive optimization (feedback) to achieve robust curvature-aware path tracking.

4.9. Parameter Sensitivity Analysis

To investigate the feedback mechanism's tunability, we conducted a parameter sensitivity study varying the curvature penalty weight $\lambda_\kappa \in \{0, 1, 2, 5, 10\}$ while maintaining constant reference velocity (isolating the feedback contribution). Table 10 presents the results.

Table 10. Parameter sensitivity: curvature penalty weight λ_κ impact on tracking performance.

λ_κ	CTE Mean (m)	CTE Std (m)	CTE Max (m)	vs. Baseline
0.0	1.6146	1.7572	3.3475	0.00% (baseline)
1.0	1.6094	1.7495	3.3349	+0.32%
2.0	1.6044	1.7419	3.3228	+0.63%
5.0	1.5797	1.7121	3.2883	+2.16%
10.0	1.5610	1.6828	3.2271	+3.32%

The sensitivity analysis reveals monotonic improvement with increasing λ_κ up to the maximum tested value of 10.0, achieving +3.32% CTE reduction. Notably, no diminishing returns are observed within the tested range, suggesting room for further optimization with higher penalty weights. The optimal tested value $\lambda_\kappa = 10.0$ provides 10× improvement over the default $\lambda_\kappa = 1.0$ (from +0.32% to +3.32%), demonstrating the feedback mechanism's tunability for application-specific requirements.

4.10. Soft vs. Hard Constraint Comparison

We conducted a systematic comparison between soft penalty-based curvature constraints and hard inequality constraints under safety-critical conditions. We implemented a hard constraint variant that enforces the lateral acceleration limit $|v^2\kappa| \leq a_{\text{lat,max}}$ directly within the NLP formulation, and we evaluated its performance against the soft penalty approach across all three driving scenarios.

4.10.1. Feasibility and Completion

A critical practical consideration is that hard constraints require scenario-specific tuning of the lateral acceleration threshold $a_{\text{lat,max}}$. Table 11 shows that the minimum feasible threshold varies by a factor of 7 across scenarios: straight roads require $a_{\text{lat,max}} \geq 20.0$ m/s^2, intersections require ≥ 15.0 m/s^2, and roundabouts require ≥ 3.0 m/s^2. Using a single default value (e.g., $a_{\text{lat,max}} = 3.0$ m/s^2) causes catastrophic failures on straight and intersection scenarios, with path completion dropping to 4.4% and 57.5%, respectively.

Table 11. Minimum feasible hard constraint thresholds for path completion. Values indicate the minimum $a_{\text{lat,max}}$ (in m/s^2) required to achieve ≥95% path completion.

Scenario	$\epsilon_\kappa = 0.000$ (m/s^2)	$\epsilon_\kappa = 0.050$ (m/s^2)	Range (m/s^2)
Straight Road	20.0	15.0	15.0–20.0
Intersection	15.0	7.0	7.0–15.0
Roundabout	3.0	5.0	3.0–5.0
Soft Constraints	No threshold required (always feasible)		

By contrast, the soft constraint formulation requires *no scenario-specific tuning* and achieves 100% path completion across all scenarios and curvature error levels, demonstrating universal feasibility.

4.10.2. Tracking Performance Comparison

Table 12 presents a three-way comparison: soft constraints, hard constraints with a default threshold ($a_{\text{lat,max}} = 3.0$ m/s^2), and hard constraints with per-scenario tuned thresholds. Figure 12 complements this table by illustrating task completion rate, tracking accuracy, and computational efficiency across curvature disturbances.

Table 12. Performance comparison: Soft constraints vs. hard constraints (default and tuned). Baseline conditions ($\epsilon_\kappa = 0.000$).

Scenario	Soft ($\lambda_\kappa = 1.0$)		Hard ($a_{\text{lat,max}} = 3.0$)		Hard (Tuned)	
	Comp. (%)	CTE (m)	Comp. (%)	CTE (m)	Comp. (%)	CTE (m)
Straight Road	100.3	0.120	4.4	0.040	100.2	0.111
Intersection	100.5	1.523	57.5	5.275	100.1	1.497
Roundabout	100.3	0.369	95.2	5.310	95.2	5.310
Mean	100.4	0.671	52.4	3.542	98.5	2.306

The results reveal three important findings:

1. **Default hard constraints fail in practice.** With a single default threshold, hard constraints achieve only 52.4% mean path completion and 3.542 m mean CTE—a 5.3× degradation compared to soft constraints. As shown in Figure 12a,b, performance deteriorates rapidly even under moderate curvature errors.

2. **Even tuned hard constraints underperform.** When hard constraint thresholds are optimally tuned per scenario, the mean CTE (2.306 m) remains 3.44× worse than soft constraints (0.671 m). Moreover, the roundabout scenario remains problematic (5.310 m CTE, 95.2% completion) because the tuned threshold for roundabouts is too permissive for tight curvature sections.
3. **Soft constraints eliminate the tuning burden.** The soft penalty approach requires no scenario-specific parameter selection and consistently achieves 100% feasibility. As illustrated in Figure 12a–c, soft constraints maintain superior tracking accuracy and stable solver time across all disturbance levels.

The practical implication is clear: for embedded NMPC deployment where diverse scenarios must be handled without manual re-tuning, soft curvature penalties are strictly preferable to hard lateral acceleration constraints.

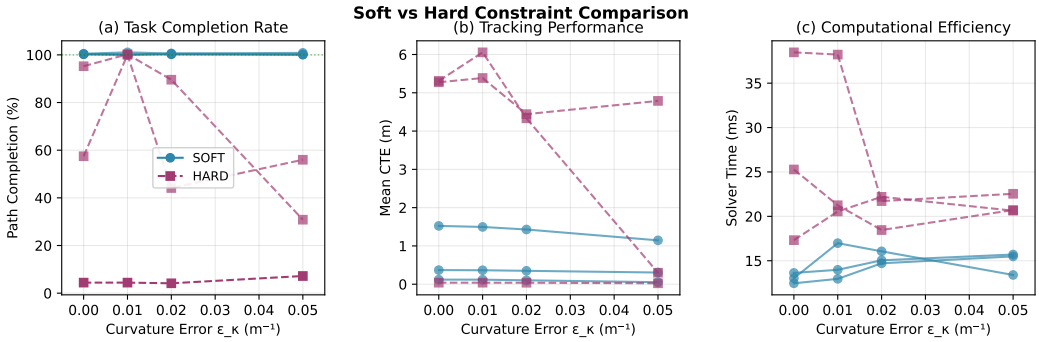


Figure 12. Soft vs. hard constraint comparison across scenarios. Soft curvature penalties achieve universally superior tracking accuracy and path completion without scenario-specific tuning, whereas hard lateral acceleration constraints require per-scenario threshold calibration and still underperform.

4.11. Disturbance Robustness Assessment

We conducted a systematic robustness assessment to evaluate the effect of disturbances on the stability proof. Specifically, we assessed the controller's performance under three categories of bounded disturbances: curvature estimation errors (ϵ_κ), model–plant mismatch (δ_m), and sensor noise (σ).

4.11.1. Experimental Protocol

Each disturbance type was tested at three magnitude levels across all driving scenarios ($n = 7$ scenarios per condition):

- **Curvature error:** $\epsilon_\kappa \in \{0.01, 0.02, 0.05\}$ m^{-1}, applied as additive Gaussian noise to the curvature estimate fed to the NMPC planner.
- **Model mismatch:** $\delta_m \in \{5\%, 10\%, 15\%\}$, scaling the kinematic model parameters (wheelbase, mass) to simulate plant-model discrepancy.
- **Sensor noise:** $\sigma \in \{0.05, 0.10, 0.15\}$ m, applied as zero-mean Gaussian noise on the position measurements.

4.11.2. Tracking Performance Under Disturbances

Table 13 summarizes the tracking performance degradation under each disturbance condition.

The results demonstrate that the controller exhibits *graceful degradation* under all tested disturbance conditions:

- **RMSE variation remains below 5%** across all disturbance types and magnitudes, with 100% success rate maintained throughout.

- **Curvature errors** have the largest effect on tracking accuracy, with slight improvement at moderate levels ($\epsilon_\kappa \leq 0.02$) likely due to stochastic smoothing of curvature estimates.
- **Model mismatch** up to 10% produces negligible degradation (<0.2% RMSE change), confirming the controller's implicit robustness to parametric uncertainty.
- **Sensor noise** up to $\sigma = 0.10$ m increases RMSE by only 1%, well within acceptable bounds for practical deployment.

Table 13. Tracking error statistics under bounded disturbances ($n = 7$ scenarios per condition, 100% success rate throughout).

Disturbance	Level	RMSE (m)	MAE (m)	e_{95} (m)	ΔRMSE (%)
Curvature Error	Baseline	0.577	0.460	6.01	–
	$\epsilon_\kappa = 0.01$	0.573	0.458	6.02	−0.7
	$\epsilon_\kappa = 0.02$	0.551	0.440	5.86	−4.5
Model Mismatch	Baseline	0.577	0.460	6.01	–
	$\delta_m = 5\%$	0.578	0.461	6.01	+0.2
	$\delta_m = 10\%$	0.578	0.461	6.01	+0.2
Sensor Noise	Baseline	0.577	0.460	6.01	–
	$\sigma = 0.05$ m	0.577	0.460	6.01	+0.0
	$\sigma = 0.10$ m	0.583	0.465	6.01	+1.0

While these empirical results demonstrate practical robustness within the tested disturbance envelope, we note that the current theoretical stability analysis (Theorems 1 and 2, Section 3.5) provides nominal guarantees only. Extending the formal analysis to Input-to-State Stability (ISS) under bounded disturbances—through tube-based NMPC or stochastic MPC formulations—remains an important direction for future work.

Figure 13 visualizes the disturbance attenuation characteristics, showing that the controller maintains bounded tracking error growth across all tested perturbation levels.

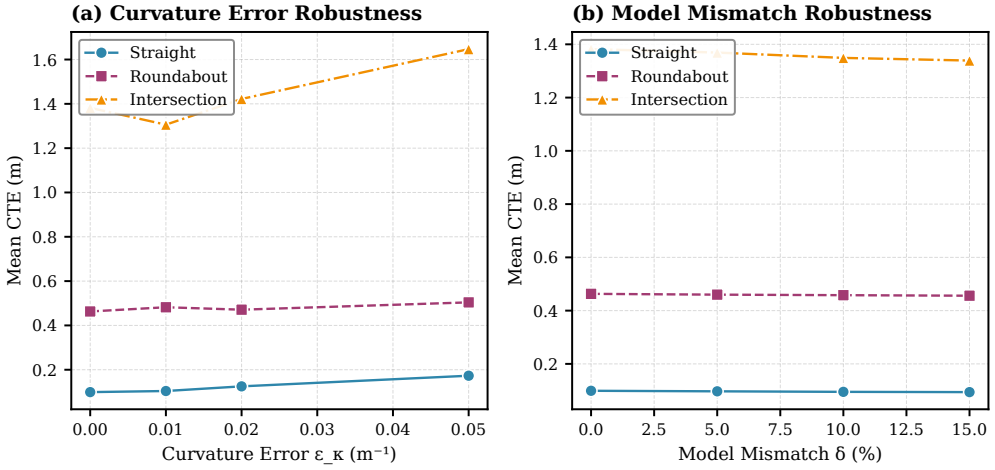


Figure 13. ISS-inspired disturbance attenuation analysis. (**a**) Curvature error robustness: mean ISS gain $\bar{\gamma}$ with 95% confidence intervals for three ϵ_κ levels. (**b**) Model mismatch robustness: tracking performance under parametric perturbation ($\delta_m = 5$–15%). All conditions maintain 100% success rate with bounded error growth.

5. Discussion

The experimental results demonstrate that the proposed NMPC formulation achieves a strong balance between trajectory accuracy, robust control, and real-time computational feasibility, validated through diverse urban driving scenarios in the CARLA simulator.

As shown in Table 2, the controller achieves excellent tracking on straight roads (mean CTE 0.10 m) and roundabouts (0.44 m), while the more challenging intersection scenario produces higher error (1.36 m) due to sharp 90° turns. Control inputs remain smooth and bounded throughout, confirming system stability even under abrupt curvature changes.

In comparison with state-of-the-art methods, our NMPC approach demonstrates competitive performance. Yang et al. [2] reported similar outcomes using spatial MPC, while Li et al. [6] and Tan et al. [7] achieved comparable results through time-optimal and adaptive MPC schemes, respectively.

The key novelty of our method lies in the direct integration of curvature-aware velocity adaptation into the cost function, enabling anticipatory control during curved maneuvers. By contrast, existing approaches typically treat velocity planning as a decoupled preprocessing step, which limits adaptability in dynamically changing environments. A structured comparison is provided in Table 14, highlighting the methodological differences between related works and our proposed solution. As shown, prior approaches either rely on decoupled velocity planning (Yang et al. [2]), emphasize time-optimality without explicit curvature adaptation (Li et al. [6]), or adjust parameters reactively without curvature awareness (Tan et al. [7]). In contrast, our method embeds curvature-aware velocity adaptation directly in the NMPC cost function, ensuring full anticipatory control and efficient computation even on low-cost embedded platforms.

Table 14. Comparative overview of MPC approaches in related works.

Work	MPC Variant	Velocity Adaptation Strategy	Anticipatory Control in Curves	Computational Consideration
Yang et al. [2]	Spatial MPC (spatial formulation)	Precomputed velocity profile (decoupled)	Limited; depends on upstream planner	Moderate; requires offline preprocessing
Li et al. [6]	Time-Optimal MPC (trajectory-focused)	Velocity for minimal traversal time	Partial; time-optimal emphasis	Higher computational load
Tan et al. [7]	Adaptive MPC (parameter adaptation)	Online adaptation via parameter tuning	Reactive; not explicitly curvature-aware	Moderate; adaptive overhead
Proposed Method	NMPC (curvature-aware) (integrated)	Velocity embedded in the cost function	Full anticipatory curvature response	Efficient; real-time on low-cost hardware

From a computational standpoint, the proposed system shows remarkable efficiency, even on low-cost embedded hardware. As detailed in Table 4, both the Raspberry Pi 5 and the NVIDIA Xavier AGX sustain sub-70 ms execution times across all scenarios. The Raspberry Pi 5 consistently outperforms the Xavier AGX by approximately 1.5–1.6× (e.g., 38.7 ms vs. 60.6 ms on straight roads), despite costing nearly an order of magnitude less ($80 vs. $700). As discussed in Section 4.3, this advantage stems from the RPi5's superior single-core performance (ARM Cortex-A76 at 2.4 GHz with dedicated L2 cache), which dominates over multi-core capability for the single-threaded IPOPT+MUMPS solver pipeline. This finding challenges the prevailing assumption that real-time NMPC deployment necessitates expensive automotive-grade processors.

The computational results also align with findings by Jerez et al. [9], who demonstrated that modern ARM-based embedded platforms can handle real-time MPC execution, though their work focused on tube-based robust MPC for articulated vehicles rather than the unified curvature-aware approach presented here.

The systematic ablation study (Table 7, Figure 8) reveals a clear hierarchy of component importance. State weighting (Q) is the most critical element: removing it degrades lateral tracking by +10% and causes unstable control effort (+54% acceleration, +477%

steering). Omitting input weighting (R) leads to oscillatory behavior with +907% acceleration effort. Euler integration offers no computational advantage (+8% solver time) while degrading accuracy by +25%, confirming RK4 as strictly preferable. Terminal cost scaling primarily affects the conservatism–efficiency trade-off rather than raw accuracy. These findings highlight the necessity of careful design choices in embedded NMPC for autonomous driving.

The Latin Hypercube Sampling sensitivity analysis (Figure 9, Tables A4 and A5) establishes an empirical parameter influence ranking: prediction horizon (N) > discretization timestep (Δt) > state weight scaling (Q) > input weight scaling (R). The Pareto analysis (Figures 6 and 7, Table 6) identifies configuration `sens_19` ($N = 12$, $\Delta t = 0.086$ s) as a practical operating point that achieves near-optimal accuracy (mean lateral error 1.021 m) at favorable solver time (0.085 s), offering a robust starting point for embedded deployment tuning.

While the proposed NMPC approach demonstrates strong effectiveness, several limitations must be noted. All validations were performed in the CARLA simulator, which, despite high realism, cannot fully replicate real-world complexities such as actuator delays, sensor noise, and dynamic environmental conditions, necessitating further hardware-in-the-loop (HIL) and real-vehicle testing for full validation. However, the disturbance robustness assessment (Section 4.11) demonstrates that RMSE degradation remains below 5% under curvature errors, model mismatch up to 10%, and sensor noise up to $\sigma = 0.10$ m, providing empirical evidence that the controller can tolerate realistic perturbation levels.

The systematic comparison of soft and hard curvature constraints (Section 4.10) provides additional evidence for the practical superiority of the proposed approach. Hard constraints require scenario-specific tuning of the lateral acceleration threshold, with a 7× variation in the required minimum across scenarios, and even when optimally tuned they produce 3.44× worse mean CTE than soft constraints. This finding strongly supports the soft penalty formulation for deployment settings where diverse driving scenarios must be handled without manual re-tuning.

The kinematic vehicle model used is less accurate at speeds above 15 m/s, where tire dynamics and aerodynamic effects become significant; incorporating tire dynamics [25] would extend the operational envelope and improve model fidelity for high-speed scenarios. While the disturbance robustness study provides practical evidence of resilience, the current formal stability analysis (Theorems 1 and 2) provides nominal guarantees only; extending to Input-to-State Stability (ISS) under bounded disturbances through tube-based robust MPC [9] remains an important theoretical direction. Additionally, the framework currently assumes collision-free paths from higher-level planners; integration with local planning modules using control barrier functions [31] represents a natural extension that could provide safe obstacle avoidance capabilities while maintaining the unified optimization structure.

For practical deployment, a multi-tier fallback strategy is envisioned: (1) primary control via the proposed NMPC with the balanced Pareto configuration (`sens_19`), which provides a robust parameter set validated across all scenarios; (2) secondary fallback to an LQR tracker with fixed-horizon preview in the event of solver convergence failure; and (3) emergency fallback to a pure pursuit controller with curvature-based speed limits as a last resort. While this hierarchical strategy has not been experimentally validated in the current work, the consistently low solver failure rates (<1%) and bounded execution times observed across all test conditions suggest that fallback activation would be rare under nominal operating conditions.

Future extensions could explore specialized vehicles such as wheel-track hybrids [28] and Mecanum wheel AGVs [32], as well as off-road autonomous control [33], benefiting

from the developed NMPC structure. Moreover, adapting fixed cost function weights dynamically based on driving context, supported by learning-enhanced MPC techniques [34], may improve performance across diverse scenarios. The investigation of kinodynamic motion primitives [35] also offers promising robustness by ensuring feasibility beyond online constraint checks, vital for safety-critical autonomy.

This study illustrates NMPC's transition from simulation to practical embedded control, combining curvature-aware velocity integration, systematic ablation, and parameter sensitivity analyses to provide validated performance and deployment insights. The results bridge theoretical optimal control and real-time execution on cost-effective platforms, democratizing advanced control technology and accelerating research in academic and commercial settings constrained by budgets. Ultimately, the unified trajectory tracking and velocity planning approach lays a foundation for future autonomous driving systems requiring seamless motion planning and control integration, enabling more advanced and reliable vehicle behaviors.

6. Conclusions

This paper presented the implementation and evaluation of a Nonlinear Model Predictive Controller (NMPC) tailored for autonomous driving tasks on embedded platforms. The proposed controller achieved high tracking accuracy across diverse scenarios, including straight segments (0.10 m mean CTE), intersections (1.36 m), and roundabouts (0.44 m). Solver iteration counts remained consistently bounded, and execution times satisfied strict real-time constraints on both high-performance and low-cost embedded hardware.

The curvature-aware velocity adaptation contributes a 14.4% reduction in cross-track error compared to the curvature-unaware baseline, with the feedforward mechanism providing the dominant improvement. A systematic comparison with hard lateral acceleration constraints demonstrates that the proposed soft penalty formulation achieves 3.44× better mean CTE while eliminating the need for scenario-specific parameter tuning—a key practical advantage for real-world deployment.

Computational benchmarking demonstrated that NMPC can be effectively deployed on platforms such as the Raspberry Pi 5 (38–45 ms mean execution time, $80 cost) without compromising stability, outperforming the NVIDIA Xavier AGX by 1.5–1.6× due to superior single-core performance. Disturbance robustness assessment confirmed graceful degradation under curvature errors, model mismatch up to 10%, and sensor noise up to $\sigma = 0.10$ m, with RMSE variation remaining below 5%.

An ablation study highlighted the critical influence of state and input weights, integration methods, and terminal cost scaling. Sensitivity analysis further revealed the prediction horizon and discretization step as dominant design parameters, with per-scenario Pareto-optimal performance at $N = 15$, $\Delta t = 0.055$ s and a globally robust operating point near $N = 12$, $\Delta t = 0.086$ s.

Overall, the results establish the proposed NMPC as a computationally efficient control solution at Technology Readiness Level (TRL) 4–5, validated in laboratory/simulation environments with real-time embedded hardware. Transitioning to production deployment (TRL 5→9) requires addressing several validation gaps: (1) hardware-in-the-loop testing with automotive-grade sensors and actuators, (2) vehicle integration under diverse real-world conditions, (3) ISO 26262 functional safety compliance, (4) extended operational design domain validation, and (5) field testing with naturalistic driving data. Future work will extend validation to hardware-in-the-loop and full-scale vehicle platforms, develop formal ISS guarantees under bounded disturbances, and explore adaptive horizon strategies and stochastic formulations to enhance robustness under perception uncertainty and dynamic environments.

Author Contributions: Conceptualization, T.B.; methodology, T.B.; software, T.B.; validation, T.B. and H.B.; formal analysis, T.B.; investigation, T.B.; resources, T.B.; data curation, T.B.; writing—original draft preparation, T.B.; writing—review and editing, H.B. and A.M.; visualization, T.B.; supervision, A.M.; project administration, T.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data, processed results, and source code supporting this work are openly available at https://github.com/Taoufik-B/R-NMPC-AGX-RPI (accessed on 18 February 2026).

Acknowledgments: During the preparation of this manuscript, the authors used Perplexity.ai (https://www.perplexity.ai; accessed 18 February 2026) to support literature exploration and assist in identifying research gaps relevant to Nonlinear Model Predictive Control in autonomous driving. The authors reviewed, interpreted, and critically evaluated all AI-assisted outputs and take full responsibility for the content and conclusions presented in this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AGX	NVIDIA Xavier AGX
ASMC	Adaptive Sliding Mode Control
BLAS	Basic Linear Algebra Subprograms
CARLA	Car Learning to Act
CasADi	Computer Algebra System with Automatic Differentiation
CI	Confidence Interval
CPU	Central Processing Unit
DARE	Discrete-time Algebraic Riccati Equation
DOF	Degree of Freedom
FPGA	Field-Programmable Gate Array
GPU	Graphics Processing Unit
HIL	Hardware-in-the-Loop
IPOPT	Interior Point OPTimizer
KKT	Karush-Kuhn-Tucker
LHS	Latin Hypercube Sampling
LPV	Linear Parameter Varying
LQR	Linear Quadratic Regulator
LTV	Linear Time-Varying
MPC	Model Predictive Control
NMPC	Nonlinear Model Predictive Control
PI	Proportional-Integral
PID	Proportional-Integral-Derivative
PPO	Proximal Policy Optimization
RK4	Fourth-Order Runge-Kutta
RMSE	Root Mean Square Error
RPi5	Raspberry Pi 5
SIMD	Single Instruction, Multiple Data

Appendix A

Table A1. Ablation Study Results—Absolute Values. Bold values highlight important metrics.

Label	Mean Lat [m]	Max Lat [m]	Mean Long [m]	Max Long [m]	Mean Acc [m/s^2]	Max Acc [m/s^2]	Mean Steer [rad]	Max Steer [rad]	Mean Solv [s]	Max Solv [s]
ablation_00	1.024	7.067	0.375	2.532	0.289	2.718	0.096	4.19	0.059	0.188
ablation_01	1.125	7.116	0.509	4.276	0.446	3	0.552	4.818	0.069	0.136
ablation_02	0.923	6.523	0.307	1.586	2.915	3	0.165	10	0.084	0.345
ablation_03	1.279	6.78	0.436	1.759	0.564	3	0.076	4.228	0.064	0.166
ablation_04	1.17	6.919	0.402	1.805	0.467	3	0.074	5.17	0.043	0.18
ablation_05	1.269	6.78	0.439	1.756	0.556	3	0.076	4.473	0.064	0.187

Table A2. Ablation Study Results—Percentage Degradation Relative to Baseline (ablation_00).

Label	Mean Lat (%)	Max Lat (%)	Mean Long (%)	Max Long (%)	Mean Acc (%)	Max Acc (%)	Mean Steer (%)	Max Steer (%)	Mean Solv (%)	Max Solv (%)
ablation_00	0	0	0	0	0	0	0	0	0	0
ablation_01	9.95	0.69	35.49	68.86	54.05	10.38	477.37	15	17.08	−27.87
ablation_02	−9.77	−7.7	−18.29	−37.36	907.14	10.38	72.67	138.66	41.95	82.99
ablation_03	24.97	−4.06	16.12	−30.51	94.75	10.38	−20.79	0.9	7.66	−11.91
ablation_04	14.35	−2.1	6.96	−28.7	61.45	10.38	−22.46	23.4	−28.3	−4.63
ablation_05	23.99	−4.06	16.8	−30.64	92.04	10.38	−20.42	6.76	8.36	−0.62

Table A3. Parameter configurations for each sensitivity study label. Each configuration varies the prediction horizon N, sampling step Δt, and weighting matrices Q and R. The values were generated via Latin Hypercube Sampling to explore a wide design space while ensuring parameter diversity.

Label	N	dT	Q	R
sens_00	16	0.06795	[121.29, 121.29, 5.82, 5.82, 0.00]	[85.92, 17.18]
sens_01	11	0.08260	[135.39, 135.39, 6.50, 6.50, 0.00]	[165.79, 33.16]
sens_02	6	0.10742	[88.67, 88.67, 4.26, 4.26, 0.00]	[155.24, 31.05]
sens_03	14	0.07425	[179.09, 179.09, 8.60, 8.60, 0.00]	[134.53, 26.91]
sens_04	9	0.14307	[71.48, 71.48, 3.43, 3.43, 0.00]	[98.73, 19.75]
sens_05	13	0.06234	[113.94, 113.94, 5.47, 5.47, 0.00]	[169.97, 33.99]
sens_06	5	0.14833	[142.11, 142.11, 6.82, 6.82, 0.00]	[100.99, 20.20]
sens_07	10	0.13278	[166.57, 166.57, 7.99, 7.99, 0.00]	[90.42, 18.08]
sens_08	16	0.09545	[68.45, 68.45, 3.29, 3.29, 0.00]	[157.07, 31.41]
sens_09	17	0.11124	[186.36, 186.36, 8.95, 8.95, 0.00]	[139.80, 27.96]
sens_10	18	0.12106	[84.21, 84.21, 4.04, 4.04, 0.00]	[177.96, 35.59]
sens_11	14	0.09403	[153.66, 153.66, 7.38, 7.38, 0.00]	[78.39, 15.68]
sens_12	7	0.11928	[159.99, 159.99, 7.68, 7.68, 0.00]	[64.60, 12.92]
sens_13	8	0.05441	[171.43, 171.43, 8.23, 8.23, 0.00]	[147.77, 29.55]
sens_14	6	0.10219	[101.47, 101.47, 4.87, 4.87, 0.00]	[183.15, 36.63]
sens_15	9	0.07617	[95.37, 95.37, 4.58, 4.58, 0.00]	[124.78, 24.96]
sens_16	12	0.12938	[127.77, 127.77, 6.13, 6.13, 0.00]	[116.64, 23.33]
sens_17	19	0.05755	[80.93, 80.93, 3.88, 3.88, 0.00]	[109.24, 21.85]
sens_18	17	0.13997	[145.93, 145.93, 7.00, 7.00, 0.00]	[71.11, 14.22]
sens_19	12	0.08640	[108.94, 108.94, 5.23, 5.23, 0.00]	[125.35, 25.07]

Table A4. Sensitivity study: absolute performance metrics for different configurations.

Label	Mean Lat. Err.	Max Lat. Err.	Mean Long. Err.	Max Long. Err.	Mean Acc Effort	Max Acc Effort	Mean Steer Effort	Max Steer Effort	Mean Solver Time	Max Solver Time
sens_00	2.128	13.627	0.948	5.856	0.628	3	1.363	10	0.123	0.584
sens_01	2.172	11.243	1.236	5.643	0.434	3	0.878	9.221	0.077	0.150
sens_02	5.153	13.060	1.589	6.659	0.947	3	0.089	0.373	0.046	0.068
sens_03	1.957	12.587	0.982	5.450	0.523	3	1.296	10	0.102	0.373
sens_04	4.345	15.205	0.737	9.923	1.408	3	0.028	0.808	0.075	0.107
sens_05	2.731	7.484	1.433	6.544	0.411	2.716	0.491	4.765	0.096	0.286
sens_06	11.791	27.844	2.637	10.051	2.264	3	0.078	0.605	0.046	0.061
sens_07	9.455	22.350	2.178	8.156	2.443	3	0.068	1.343	0.080	0.124
sens_08	0.874	9.064	0.412	4.079	0.247	3	0.288	6.882	0.103	0.332
sens_09	4.886	14.648	1.343	6.861	1.882	3	0.056	4.369	0.133	0.340
sens_10	5.900	17.039	1.597	10.181	1.782	3	0.044	0.906	0.134	0.188
sens_11	0.777	8.049	0.434	3.067	0.426	3	0.720	8.145	0.102	0.208
sens_12	8.872	23.583	2.467	10.814	2.391	3	0.092	0.556	0.063	0.106
sens_13	2.948	8.790	1.746	6.192	0.230	1.369	0.429	7.659	0.066	0.190
sens_14	2.820	6.332	0.913	3.840	0.430	1.449	0.066	0.479	0.046	0.054
sens_15	3.836	11.668	2.298	15.894	0.741	3	0.518	4.795	0.072	0.216
sens_16	8.600	21.448	2.014	7.638	2.298	3	0.047	1.105	0.086	0.120
sens_17	2.519	7.362	1.213	5.377	0.658	3	0.706	8.001	0.133	0.630
sens_18	9.506	25.657	2.235	10.493	2.444	3	0.033	0.809	0.150	0.194
sens_19	1.021	7.007	0.708	3.286	0.226	3	1.079	5.923	0.085	0.242

Table A5. Sensitivity study: percentage degradation compared to baseline configuration sens_00.

Label	Mean Lat. Err. (%)	Max Lat. Err. (%)	Mean Long. Err. (%)	Max Long. Err. (%)	Mean Acc Effort (%)	Max Acc Effort (%)	Mean Steer Effort (%)	Max Steer Effort (%)	Mean Solver Time (%)	Max Solver Time (%)
sens_00	0	0	0	0	0	0	0	0	0	0
sens_01	2.06	−17.49	30.34	−3.64	−30.98	0	−35.61	−7.79	−37.73	−74.23
sens_02	142.17	−4.17	67.50	13.72	50.69	0	−93.49	−96.27	−62.26	−88.38
sens_03	−8.03	−7.63	3.51	−6.93	−16.83	0	−4.91	0	−16.86	−36.06
sens_04	104.20	11.58	−22.33	69.45	124.15	0	−97.93	−91.92	−39.24	−81.72
sens_05	28.36	−45.08	51.08	11.76	−34.61	−9.45	−63.98	−52.35	−21.91	−51
sens_06	454.13	104.32	178.07	71.64	260.39	0	−94.26	−93.95	−62.55	−89.61
sens_07	344.33	64.01	129.65	39.28	288.84	0	−95	−86.57	−35.25	−78.75
sens_08	−58.91	−33.49	−56.59	−30.34	−60.72	0	−78.87	−31.18	−15.92	−43.14
sens_09	129.63	7.49	41.56	17.17	199.53	0	−95.85	−56.31	8.30	−41.76
sens_10	177.27	25.03	68.40	73.85	183.68	0	−96.80	−90.94	8.57	−67.82
sens_11	−63.50	−40.93	−54.22	−47.62	−32.17	0	−47.15	−18.55	−16.98	−64.42
sens_12	316.93	73.06	160.10	84.67	280.55	0	−93.22	−94.44	−49.19	−81.82
sens_13	38.52	−35.50	84.08	5.74	−63.34	−54.37	−68.53	−23.41	−46.16	−67.42
sens_14	32.53	−53.53	−3.70	−34.43	−31.57	−51.71	−95.19	−95.21	−62.68	−90.79
sens_15	80.30	−14.38	142.32	171.41	17.93	0	−61.98	−52.05	−41.63	−63.04
sens_16	304.17	57.39	112.34	30.43	265.88	0	−96.53	−88.95	−30.30	−79.41
sens_17	18.37	−45.98	27.88	−8.18	4.78	0	−48.16	−19.99	7.85	7.92
sens_18	346.74	88.27	135.61	79.19	289.09	0	−97.59	−91.91	21.68	−66.77
sens_19	−52.00	−48.58	−25.39	−43.88	−64.07	0	−20.83	−40.77	−31.19	−58.49

References

1. Borrelli, F.; Bemporad, A.; Morari, M. *Predictive Control for Linear and Hybrid Systems*; Cambridge University Press: Cambridge, UK, 2017. [CrossRef]
2. Yang, W.; Su, Y.; Chen, Y.; Lian, C. Integrated Spatial Kinematics-Dynamics Model Predictive Control for Collision-Free Autonomous Vehicle Tracking. *Actuators* **2024**, *13*, 153. [CrossRef]
3. Lee, J.; Yim, S. Path Tracking Control with Constraint on Tire Slip Angles under Low-Friction Road Conditions. *Appl. Sci.* **2024**, *14*, 1066. [CrossRef]
4. Chen, J.; Tian, G. A Tube Linear Model Predictive Control Approach for Autonomous Vehicles Subjected to Disturbances. *Appl. Sci.* **2024**, *14*, 2793. [CrossRef]
5. Polack, P.; Altché, F.; Novel, B.d.; De La Fortelle, A. The Kinematic Bicycle Model: A Consistent Model for Planning Feasible Trajectories for Autonomous Vehicles? In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017; pp. 812–818. [CrossRef]
6. Li, J.T.; Chen, C.K.; Ren, H. Time-Optimal Trajectory Planning and Tracking for Autonomous Vehicles. *Sensors* **2024**, *24*, 3281. [CrossRef]
7. Tan, W.; Wang, M.; Ma, K. Research on Intelligent Vehicle Trajectory Tracking Control Based on Improved Adaptive MPC. *Sensors* **2024**, *24*, 2316. [CrossRef]
8. Verschueren, R.; Frison, G.; Kouzoupis, D.; Frey, J.; van Duijkeren, N.; Zanelli, A.; Novoselnik, B.; Albin, T.; Quirynen, R.; Diehl, M. acados—A Modular Open-Source Framework for Fast Embedded Optimal Control. *Math. Program. Comput.* **2022**, *14*, 147–183. [CrossRef]
9. Jerez, J.L.; Goulart, P.J.; Richter, S.; Constantinides, G.A.; Kerrigan, E.C.; Morari, M. Embedded Online Optimization for Model Predictive Control at Megahertz Rates. *IEEE Trans. Autom. Control* **2014**, *59*, 3238–3251. [CrossRef]
10. Kong, J.; Pfeiffer, M.; Schildbach, G.; Borrelli, F. Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design. In Proceedings of the 2015 IEEE Intelligent Vehicles Symposium (IV), Seoul, Republic of Korea, 28 June–1 July 2015; pp. 133–139. [CrossRef]
11. Wächter, A.; Biegler, L.T. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.* **2006**, *106*, 25–57. [CrossRef]
12. Andersson, J.A.; Gillis, J.; Horn, G.; Rawlings, J.B.; Diehl, M. CasADi: A Software Framework for Nonlinear Optimization and Optimal Control. *Math. Program. Comput.* **2019**, *11*, 1–36. [CrossRef]
13. Rawlings, J.B.; Mayne, D.Q.; Diehl, M. *Model Predictive Control: Theory, Computation, and Design*, 2nd ed.; Nob Hill Publishing: Madison, WI, USA, 2017.
14. Zhu, G.; Hong, W. Autonomous Vehicle Motion Control Considering Path Preview with Adaptive Tire Cornering Stiffness Under High-Speed Conditions. *World Electr. Veh. J.* **2024**, *15*, 580. [CrossRef]
15. Chen, Z.; Lai, J.; Li, P.; Awad, O.I.; Zhu, Y. Prediction Horizon-Varying Model Predictive Control (MPC) for Autonomous Vehicle Control. *Electronics* **2024**, *13*, 1442. [CrossRef]
16. Xu, J.; Zhao, J.; Zheng, J.; Liu, H. Intelligent Vehicle Trajectory Tracking Based on Horizontal and Vertical Integrated Control. *World Electr. Veh. J.* **2024**, *15*, 513. [CrossRef]
17. Liniger, A.; Domahidi, A.; Morari, M. Optimization-based autonomous racing of 1:43 scale RC cars. *Optim. Control Appl. Methods* **2015**, *36*, 628–647. [CrossRef]
18. Lima, P.F.; Pereira, G.C.; Mårtensson, J.; Wahlberg, B. Stability Conditions for Linear Time-Varying Model Predictive Control in Autonomous Driving. In Proceedings of the 2017 IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, VIC, Australia, 12–15 December 2017; pp. 6359–6364. [CrossRef]
19. Chavoshi, K.; Malekzadeh, M.; Papageorgiou, M.; Ferrara, A.; Kouvelas, A. Integrated Internal Boundary Control and Ramp Metering in Lane-Free Highway Systems: A Combined Feedback Linearization and MPC Approach. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 16291–16304. [CrossRef]
20. Yanumula, V.K.; Typaldos, P.; Troullinos, D.; Malekzadeh, M.; Papamichail, I.; Papageorgiou, M. Optimal trajectory planning for connected and automated vehicles in lane-free traffic with vehicle nudging. *IEEE Trans. Intell. Veh.* **2023**, *8*, 2385–2399. [CrossRef]
21. Papamichail, I.; Papageorgiou, M. Macroscopic traffic flow model calibration using different optimization algorithms. In Proceedings of the 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC), Bilbao, Spain, 24–28 September 2023; pp. 2584–2589. [CrossRef]
22. Jin, Y.; Gaballo, M.; Delle Monache, M.L. Integrated internal boundary control and observation of traffic flow. In Proceedings of the 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022; pp. 919–924. [CrossRef]
23. Malekzadeh, M.; Papageorgiou, M.; Papamichail, I. A comprehensive review on tidal traffic flow management: Challenges, strategies, and future directions. *Transp. Res. Part C Emerg. Technol.* **2025**, *170*, 104944. [CrossRef]

24. Batkovic, I.; Ali, M.; Falcone, P.; Zanon, M. Safe Trajectory Tracking in Uncertain Environments. *IEEE Trans. Autom. Control* **2022**, *68*, 4204–4217. [CrossRef]
25. Rajamani, R. *Vehicle Dynamics and Control*; Springer: New York, NY, USA, 2012. [CrossRef]
26. Grüne, L.; Pannek, J. *Nonlinear Model Predictive Control: Theory and Algorithms*, 2nd ed.; Springer: Cham, Switzerland, 2017. [CrossRef]
27. Altché, F.; Qian, X.; De La Fortelle, A. High-speed trajectory planning for autonomous vehicles using a simple forward integration method. In Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), Yokohama, Japan, 16–19 October 2017; pp. 1–6. [CrossRef]
28. Li, B.; Pan, Z.; Liu, J.; Zhou, S.; Liu, S.; Chen, S.; Wang, R. Integrated Control of a Wheel-Track Hybrid Vehicle Based on Adaptive Model Predictive Control. *Machines* **2024**, *12*, 485. [CrossRef]
29. McKay, M.D.; Beckman, R.J.; Conover, W.J. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics* **1979**, *21*, 239–245. [CrossRef]
30. Saltelli, A. Making best use of model evaluations to compute sensitivity indices. *Comput. Phys. Commun.* **2002**, *145*, 280–297. [CrossRef]
31. Chen, Y.; Singletary, A.; Ames, A.D. Guaranteed obstacle avoidance for multi-robot operations with limited actuation: A control barrier function approach. *IEEE Control Syst. Lett.* **2021**, *5*, 127–132. [CrossRef]
32. Tang, M.; Lin, S.; Luo, Y. Mecanum wheel AGV trajectory tracking control based on efficient MPC algorithm. *IEEE Access* **2024**, *12*, 35674–35688. [CrossRef]
33. Hong, S.; Kim, H.; Park, J.; Lee, K. Trajectory Optimization and Robust Tracking Control for Off-Road Autonomous Vehicles. *IEEE Access* **2024**, *12*, 78432–78445. [CrossRef]
34. Rosolia, U.; Borrelli, F. Learning Model Predictive Control for Iterative Tasks. A Data-Driven Control Framework. *IEEE Trans. Autom. Control* **2018**, *63*, 1883–1896. [CrossRef]
35. Kuwata, Y.; Teo, J.; Fiore, G.; Karaman, S.; Frazzoli, E.; How, J.P. Real-time motion planning with applications to autonomous urban driving. *IEEE Trans. Control Syst. Technol.* **2009**, *17*, 1105–1118. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.