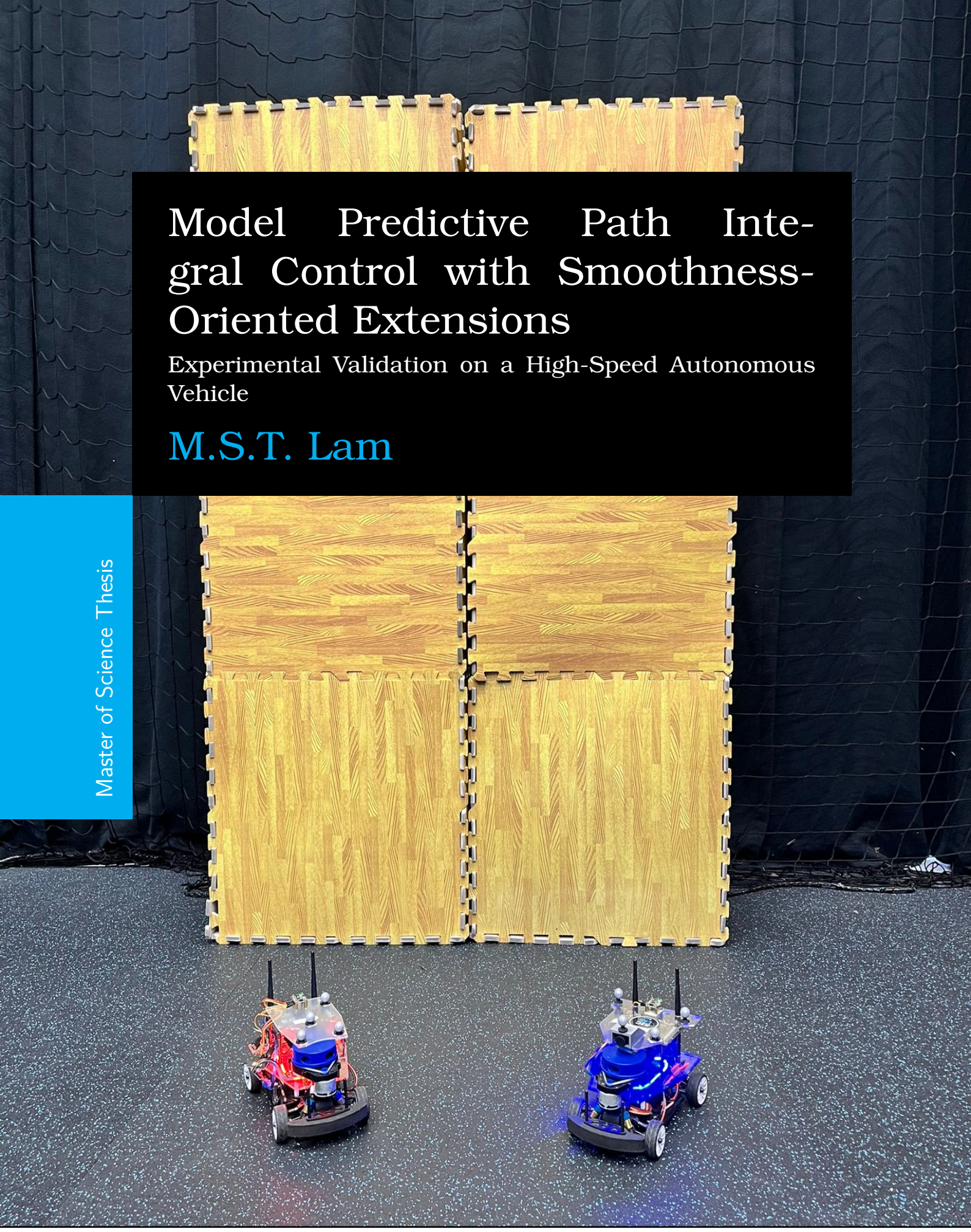
# Model Predictive Path Integral Control with Smoothness-Oriented Extensions

## Experimental Validation on a High-Speed Autonomous Vehicle

M.S.T. Lam

Master of Science Thesis



TU Delft Delft University of Technology

Delft Center for Systems and Control

# Model Predictive Path Integral Control with Smoothness-Oriented Extensions

## Experimental Validation on a High-Speed Autonomous Vehicle

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

M.S.T. Lam

February 17, 2026

Faculty of Mechanical Engineering (ME) · Delft University of Technology


# Abstract

Autonomous driving near the handling limits of a vehicle places stringent demands on the control layer, where nonlinear dynamics, actuator delay, and model mismatch dominate closed-loop behavior. Sampling-based optimal control methods, and in particular Model Predictive Path Integral (MPPI), offer an attractive alternative to deterministic receding-horizon control through stochastic rollouts and importance weighting. While MPPI has demonstrated strong performance in simulation, its real-world behavior at high speed remains insufficiently characterized.

This thesis investigates the deployment of baseline MPPI and three smoothness-oriented extensions, Dynamic Covariance MPPI, Smooth MPPI (SMPPI), and Low-pass Filtered Sampling MPPI (LFS-MPPI), on a small-scale autonomous racing platform. Experiments focus on high-speed trajectory tracking under tight lane constraints and obstacle interactions, where steering bandwidth limitations and actuation delay significantly influence stability.

Results show that baseline MPPI achieves competitive speeds and low nominal tracking error, but generates excessive high-frequency steering activity that induces sustained oscillations on hardware. Stability improvements are consistently associated with reduced steering-rate excitation. Among the evaluated extensions, LFS-MPPI provides the most favorable trade-off between smoothness, tracking accuracy, and robustness by shaping the sampling process rather than post-processing the control signal.

The experiments further reveal a fundamental coupling between model fidelity and sampling efficiency under strict real-time constraints. By increasing the proportion of dynamically plausible rollouts at reduced sample counts, LFS-MPPI enables stable deployment of a dynamic bicycle prediction model and reliable tracking up to 3.5 m/s on the physical racetrack. These findings demonstrate that sampling structure is a decisive factor in achieving stable, high-performance, real-world MPPI control.

# Table of Contents

# Acknowledgements

This thesis was carried out as part of my Master of Science graduation project at Delft University of Technology. The initial inspiration for this work arose from an early discussion with dr.ir. R.M.G. Ferrari who introduced me to high-speed MPPI-controlled autonomous racing. That demonstration sparked the ambition to pursue a real-world implementation of high-performance MPPI control, an ambition that ultimately shaped the direction of this thesis.

First and foremost, I would like to express my sincere gratitude to my daily supervisor, dr.ir. W. Martens for his continuous support, critical feedback, and trust throughout this project. Beyond our technical discussions, I have learned a great deal from his structured approach to problem solving, research planning, and perseverance in the face of uncertainty. His recurring reminder that practical implementation would surface its own challenges proved to be entirely correct, and invaluable.

I also thank my supervisor dr.ir. R.M.G. Ferrari for his rigorous and thoughtful questions, which consistently challenged me to verify assumptions and strengthen the theoretical foundations of my work. Being required to revisit derivations and modeling decisions several times ultimately resulted in a more robust and well-grounded thesis.

Furthermore, I would like to thank dr.ir. L. Lyons for introducing me to his PhD work on the DART platform and for remaining available for discussions even after his departure from TU Delft. His insights were instrumental in enabling progress toward a practical implementation.

I would also like to thank prof.dr.ir. J. Alonso-Mora and dr.ir. A.Dabiri for serving on the examination committee and for their time and careful evaluation of this work.

On a personal note, I would like to thank my friends, family, and girlfriend for their continued support and patience throughout this demanding period. Their understanding during phases where academic deadlines coincided with various logistical undertakings, some more physically demanding than others, has meant a great deal to me. Your encouragement, perspective, and occasional insistence on stepping away from the screen were invaluable.

Delft University of Technology M.S.T. Lam
February 17, 2026

# Chapter 1

# Introduction

Autonomous driving systems (ADSs) have gained significant attention due to their potential to reduce property damage, injuries, and traffic fatalities, while transforming transportation systems as a whole. A large fraction of traffic accidents can be attributed to human-related factors such as impaired driving, distraction, excessive speed, and fatigue. Consequently, the deployment of Autonomous Vehicles (AVs) is widely expected to reduce traffic-related incidents [44].

An important catalyst for research in autonomous driving was the U.S. Defense Advanced Research Projects Agency (DARPA) Grand Challenge in 2004, which stimulated both academic and industrial efforts to develop complete autonomous vehicle systems [37]. Since then, research has increasingly focused on non-holonomic ground vehicles, as these platforms closely resemble real-world road vehicles and introduce additional control challenges due to non-integrable motion constraints [45]. To formalize this gradual progression toward full autonomy, the Society of Automotive Engineers defines a taxonomy of driving automation levels, ranging from driver assistance to fully autonomous operation, emphasizing that autonomous driving is a continuum rather than a single technological breakthrough [1].

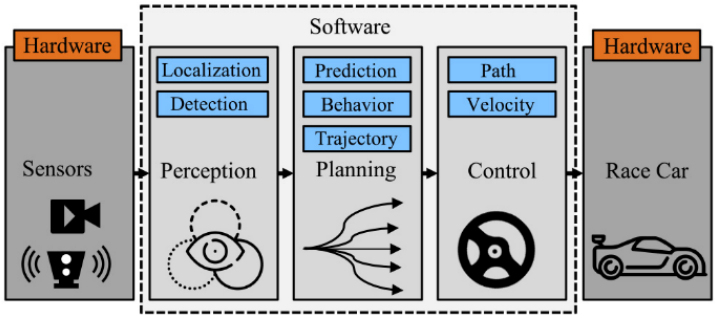



**Figure 1-1:** Layered architecture of an autonomous driving system, illustrating the decomposition into different modules [8].

The software architecture of an ADS is commonly structured either as a multi-layer pipeline or as an end-to-end system. End-to-end approaches directly map sensor data to control commands using a single learned model, whereas multi-layer architectures decompose the system into perception, planning, and control modules (Figure 1-1). While this modular structure may introduce latency and error accumulation across layers, it enables targeted development and systematic performance analysis [64]. Regardless of the chosen architecture, overall system performance ultimately depends on the reliable computation and execution of planned motions on a physical vehicle. Control therefore plays a central role in ADSs, particularly in operating regimes where vehicle dynamics and physical constraints dominate system behavior. For this reason, the control layer forms the primary focus of the thesis.

Traditional path-tracking methods (Figure 1-2), including geometric controllers such as Pure Pursuit or the Stanley controller, provide computationally efficient solutions and perform well under moderate speeds and nominal operating conditions. An effective path-tracking controller must satisfy four key requirements: a broad operating range across speeds and cornering conditions, robustness to parameter variations and disturbances, predictable tunability, and real-time feasibility on available hardware [48]. However, as vehicle speed increases and tire forces approach their physical limits, controllers often struggle to maintain robustness and tracking performance [42]. These limitations have motivated the development of advanced model-based control strategies that explicitly account for vehicle dynamics, constraints, and nonlinear behavior.

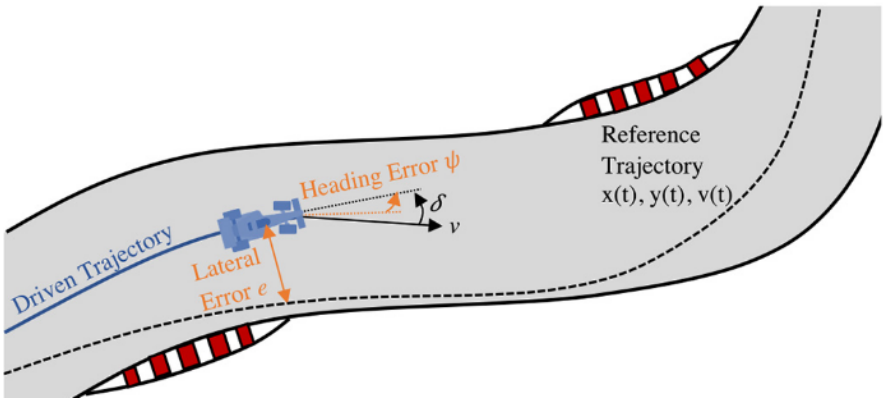



**Figure 1-2:** Path-tracking objective: the vehicle aims to follow a predefined reference trajectory (racing line) [8].

Despite steady progress in autonomous driving, achieving reliable performance across the full spectrum of driving conditions remains an open challenge. Scenarios involving high speeds, aggressive maneuvers, or operation near the physical limits of the vehicle are especially demanding [48]. In these regimes, nonlinear vehicle dynamics, actuator limitations, external disturbances, and modeling uncertainties are amplified, placing stringent requirements on the control layer. High-speed autonomous driving, and autonomous racing in particular, has therefore emerged as a valuable research domain for studying control under extreme conditions. Rather than being an end in itself, racing serves as a systematic means of exposing control algorithms to the most challenging operating regimes they may encounter. Insights obtained in this context have been shown to transfer to passenger vehicles, for example by improving trajectory tracking and velocity control in safety-critical scenarios [8].

While high-fidelity simulation environments are widely used for the development and evaluation of autonomous driving algorithms, simulation alone is insufficient to fully capture the complexities of real-world vehicle behavior. Physical systems exhibit effects that are difficult to model accurately, including unmodeled dynamics, sensor noise, actuation delays, hardware nonlinearities, and interactions with the environment [4]. As a result, controllers that perform well in simulation may exhibit degraded or unstable behavior when deployed on real vehicles, particularly under high-speed or near-limit conditions. Moreover, physical experiments introduce additional sources of variability, such as battery voltage fluctuations, computational load variations, and communication delays, which further complicate repeatability [26].

Experimental validation of aggressive autonomous driving on full-scale vehicles is, however, severely constrained by cost, safety, and logistical considerations. Operating vehicles near their handling limits entails a non-negligible risk of damage and requires extensive safety measures, specialized infrastructure, and significant financial investment. These constraints make systematic experimentation in extreme driving regimes impractical for most research groups and limit the amount of real-world data that can be collected in such conditions.

Scaled AV platforms provide a practical and effective alternative. By preserving the essential characteristics of vehicle dynamics while significantly reducing cost and risk, small-scale platforms enable repeatable real-world experimentation under controlled conditions. Prior work has emphasized that such platforms are particularly well suited for studying aggressive driving scenarios that are difficult to investigate either in simulation or with full-scale vehicles. In particular, planning and control for racing and platooning have been shown to benefit from small-scale experimental testbeds [26]. Scaled platforms also complement simulation-based development by exposing control algorithms to real-world imperfections that cannot be faithfully reproduced in software alone [18]. One such platform is the Delft's Autonomous-driving Robotic Testbed (DART) developed at Delft University of Technology [33]. Designed specifically for autonomous high-speed research, DART provides a modern small-scale testbed that bridges simulation and real-world experimentation, making it a suitable platform for evaluating advanced racing controllers.

Within this context, the thesis focuses on the control of a single non-holonomic AV operating at high speed on a small-scale experimental platform. The emphasis is placed on trajectory tracking near the limits of performance, where nonlinear vehicle behavior becomes prominent and control robustness is critical. Rather than addressing complex urban environments with highly dynamic traffic interactions, the scope is narrowed to isolate and study the fundamental control challenges that arise in aggressive driving scenarios under strict computational constraints.

## 1-1 Sampling-Based Control

Sampling-based control methods have gained increasing attention in recent years as a means of addressing the challenges posed by nonlinear dynamics, non-convex constraints, and limited computational budgets in real-time control. The term sampling-based control is used to refer to stochastic optimal control methods that approximate optimal policies through randomized rollouts and importance weighting. Rather than relying on the explicit solution of large-scale optimization problems, these methods approximate optimal control policies through

stochastic sampling and importance weighting, offering a flexible and scalable alternative to classical Model Predictive Control (MPC) formulations.

Path Integral (PI) control methods, in particular, have emerged as a prominent class of stochastic optimal control approaches. Recent surveys highlight their growing relevance in applications requiring fast replanning and robustness to model uncertainty [23]. Model Predictive Path Integral (MPPI) control extends these ideas to a receding-horizon framework, enabling real-time trajectory optimization for systems with nonlinear dynamics and high-dimensional state spaces. By avoiding explicit gradient computation and constraint linearization, MPPI is well suited for aggressive driving scenarios in which vehicle dynamics operate far from nominal conditions. The suitability of MPPI for high-speed autonomous driving has been demonstrated in a range of simulation and experimental studies [57, 25, 17].

At the same time, the sampling-based nature of MPPI introduces challenges that become especially pronounced in real-world implementations. Control sequences generated through stochastic sampling have been shown to exhibit significant high-frequency variations, which leads to jerky or oscillatory actuation commands. While such behavior can often be tolerated in simulation, it may excite unmodeled dynamics, increase actuator wear, and degrade stability on physical systems [11]. In this thesis, stability refers to bounded closed-loop behavior characterized by non-divergent lateral error and absence of sustained oscillatory steering behavior. These observations indicate that achieving high performance with MPPI in real-world systems requires not only accurate modeling and cost design, but also careful consideration of how control inputs are sampled, represented, and updated. Correspondingly, several MPPI variants have been proposed to incorporate smoothness directly into the control representation or sampling process, rather than relying on post-hoc filtering [25, 29].

Three such extensions are examined in the context of high-speed autonomous driving on a real-world small-scale platform: dynamic covariance adaptation [55], Smooth MPPI (SMPPI) [25], and low-pass filtered sampling [24]. The cited works provide the conceptual basis for these extensions, while their implementations are adapted to the present control architecture. Their design principles, constructed implementations and performance characteristics are systematically analyzed in the subsequent chapters.

## 1-2 Contributions

A substantial body of recent work has investigated MPPI-based control methods and a variety of algorithmic extensions, with most evaluations conducted in simulation environments [14, 6] or under constrained experimental conditions, such as low operating speeds for ground vehicles [49, 10, 61]. These studies provide important insights into the theoretical properties and qualitative behavior of sampling-based controllers. However, simulation-based validation and low-speed experiments do not fully expose the practical challenges that arise when deploying MPPI on physical systems operating near their handling limits, where actuator dynamics, sensing imperfections, and model mismatch play a dominant role.

While baseline MPPI has been successfully demonstrated on small-scale autonomous racing platforms [57], the experimental evaluation of MPPI extensions remains dispersed across application domains and operating regimes. Smooth MPPI has primarily been investigated in simulation, with an emphasis on aggressive maneuvering behavior rather than real-world

deployment [25]. Dynamic covariance adaptation methods, such as CoVO-MPPI, have been validated on physical platforms including quadrotors, but have not been studied in the context of high-speed ground vehicle racing [62]. Similarly, low-pass and low-frequency sampling strategies have been evaluated on ground vehicles or full-scale platforms, yet typically under simplified conditions such as oval tracks [24] or not focused on driving at high-speeds [54], which only partially reflect the demands of high-speed autonomous driving. The thesis addresses these gaps through a structured experimental evaluation of MPPI and selected smoothness-oriented extensions on a real-world small-scale autonomous racing platform. The focus lies on high-speed trajectory tracking in scenarios with sharp corners and strict track boundary constraints, operating in regimes where actuator limits, latency, and model mismatch significantly influence closed-loop behavior.

The primary contribution is the implementation and evaluation of SMPPI, dynamic covariance MPPI, and Low-pass Filtered Sampling MPPI (LFS-MPPI) within the context of high-speed small-scale autonomous racing. While these methods have been studied in simulation or in other application domains, their behavior in near-limit ground vehicle racing has not been systematically examined. This thesis therefore extends their evaluation to a practically demanding operating regime.

A second contribution is a controlled, side-by-side comparison of baseline MPPI and its extensions under identical architectural, tuning, and experimental conditions. By implementing all controllers within a unified framework and testing them on the same tracks, reference speeds, and boundary constraints, the study enables a direct assessment of performance trade-offs that are typically obscured by cross-study differences.

Finally, this thesis contributes to the maturation of the DART small-scale autonomous racing platform by deploying and validating advanced sampling-based controllers under high-speed laboratory conditions. Through systematic experimentation in both simulation and hardware, it demonstrates the suitability of DART as a testbed for evaluating near-limit control strategies and establishes a foundation for further research on high-performance sampling-based control.

## 1-3 Research Questions

The thesis investigates the practical deployment of MPPI control and smoothness-oriented extensions for high-speed autonomous driving on a real-world small-scale platform. The following research questions are addressed:

- How does baseline MPPI perform when deployed on a real-world small-scale autonomous racing platform operating near the limits of handling?
- Which implementation choices and modifications are required to achieve stable and reliable baseline MPPI performance on physical hardware?
- To what extent do improvements observed in simulation translate to real-world performance under identical experimental conditions?
- What trade-offs arise between control smoothness, responsiveness, and tracking accuracy in real-world MPPI-based control?

- How do smoothness-oriented MPPI extensions affect tracking performance, control stability, and robustness in high-speed racing scenarios?

## 1-4 Thesis Outline

The remainder of the thesis is organized into five chapters.

Chapter 2 introduces the theoretical foundations of the thesis. It reviews vehicle dynamics modeling, MPC, and the theoretical principles underlying MPPI control. The chapter positions MPPI within the broader landscape of nonlinear and stochastic model predictive control methods.

Chapter 3 presents the methodological framework used throughout the thesis. It details the MPPI formulation adopted in this thesis, including the cost function design, and prediction models. The simulation environment and implementation-specific considerations that originate from the real-world platform are introduced. The chapter further establishes a baseline validation of the MPPI implementation, demonstrating correct algorithmic behavior prior to comparative evaluation. Finally, the evaluation pipeline is formalized by defining the driving scenarios, performance metrics, and analysis procedures used consistently throughout the remainder of the thesis.

Chapter 4 provides a systematic comparative study of baseline MPPI and several smoothness-oriented MPPI extensions in simulation. The controllers are evaluated under progressively more demanding operating conditions, including higher reference speeds and sharp corners. Performance is analyzed with respect to tracking accuracy, control smoothness, sample efficiency, robustness, and characteristic failure modes.

Chapter 5 transfers the most relevant findings to the physical small-scale vehicle platform. Experimental results are presented under controlled laboratory conditions, and discrepancies between simulation and real-world behavior are analyzed. The chapter assesses practical deployment, thereby validating the practical applicability of the proposed extensions.

Finally, Chapter 6 synthesizes the main findings of the thesis. It reflects on the trade-offs between tracking performance and control smoothness in sampling-based control, discusses implications for real-world deployment of MPPI-type algorithms, and outlines concrete directions for future research.

# Chapter 2

# Background

The control problem considered in this thesis is finite-horizon path tracking with progress along a known reference, using a receding-horizon controller that must run in real time on embedded consumer hardware. This motivates prediction models and optimization methods that balance computational efficiency, robustness to modeling error, and control smoothness. This chapter consolidates the modeling assumptions, control formulations, and notation used throughout the thesis. It provides a compact reference that supports the controller design and experimental analysis in later chapters.

The chapter proceeds in four parts. First, the modeling scope is defined and the vehicle models used for prediction and simulation are introduced, progressing from a kinematic to a dynamic bicycle model. Second, Model Predictive Control (MPC) is reviewed, and Model Predictive Contouring Control (MPCC) is presented as a path-tracking formulation that separates tracking error into contouring and lag components. Third, Model Predictive Path Integral (MPPI) is derived and presented in the notation used throughout the remainder of the thesis, including its exponential cost-weighting update and its practical receding-horizon implementation. Finally, the MPPI extensions that target control smoothness are introduced at a conceptual level.

## 2-1 Vehicle Models

Accurate vehicle modeling is crucial for model-based controller design and consequently for an Autonomous Vehicle (AV), especially when driving at higher speeds or under aggressive maneuvers. Because real vehicle dynamics involve many nonlinear effects, most models simplify the relations by capturing only the dominant influences on overall behavior. This introduces a natural trade-off between model accuracy and computational demand. Consequently, researchers have devoted substantial effort to exploring and refining vehicle modeling approaches. This is reflected in extensive prior research on individual vehicle subsystems, such as tire dynamics [36]. Researchers have developed models of varying complexity, with each category striking a different balance between simplicity and accuracy. Purely kinematic

models occupy one end of the spectrum, deriving vehicle states from straightforward geometric relationships. At the opposite end are what studies call "fully dynamic" models, which incorporate tire slip, body roll, and load transfer dynamics [22].

### 2-1-1 Assumptions, Modeling Techniques, and Notation

Before introducing the specific vehicle models used for prediction and control, three aspects are discussed: the modeling assumptions that define the scope of the dynamics captured, the modeling techniques used to reduce the full vehicle to a control-oriented representation, and the coordinate-frame and notation conventions used throughout the thesis.

#### Modeling assumptions

The vehicle dynamics are formulated under the assumption of planar motion, meaning that roll and pitch dynamics are neglected and the vehicle is assumed to move on a flat road surface. As a result, yaw is the only rotational degree of freedom considered. Furthermore, only the front wheels are assumed to steer, which is consistent with standard passenger vehicle configurations and widely adopted in control-oriented vehicle models [3]. These assumptions are appropriate for high-speed path tracking on smooth, planar test tracks, where lateral and yaw dynamics dominate the vehicle behavior.

#### Modeling techniques

To reduce the full vehicle dynamics to a tractable form suitable for real-time predictive control, the single-track, or "bicycle," modeling approach is adopted. Under small to moderate steering angles, the left and right tires on each axle experience nearly identical slip conditions, allowing both tires on an axle to be represented by a single effective wheel, as shown in Figure 2-1. Combined with the planar-motion assumption, this yields a simplified representation in which the front and rear wheels are aligned along the vehicle centerline. The resulting bicycle geometry captures the dominant longitudinal and lateral dynamics while maintaining a favorable balance between model fidelity and computational efficiency.

#### Coordinate frames and notation

Vehicle dynamics are expressed using a body-fixed coordinate frame attached to the vehicle's center of gravity (CG) [3]. The body frame, denoted by $(x_v, y_v)$, has its origin at the CG, with the $x_v$ axis aligned with the vehicle's longitudinal direction (positive forward) and the $y_v$ axis pointing laterally to the left. Forces, velocities, and control inputs are primarily expressed in this body-fixed frame, which simplifies the equations of motion. An inertial, ground-fixed frame denoted by $(x, y)$ is introduced to describe the vehicle's global position and orientation. The yaw angle $\psi$ is defined as the angle between the inertial $x$ axis and the vehicle's longitudinal axis, measured positive in the anticlockwise direction.

The steering angle $\delta$ represents the front-wheel steering input and is defined as positive for anticlockwise rotation. The yaw rate $\omega$ denotes the rotational velocity about the vertical axis,

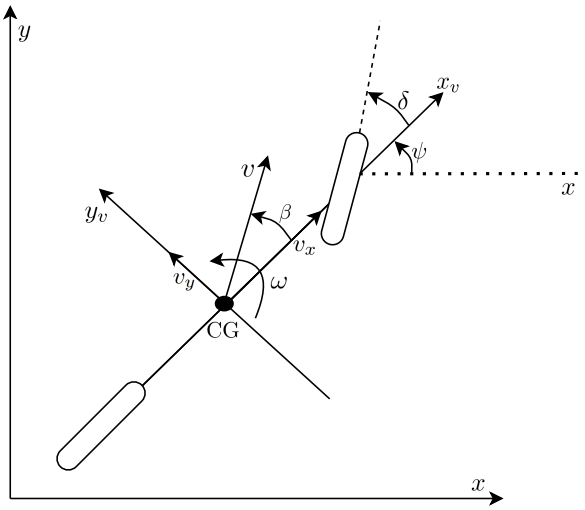



**Figure 2-1:** Single-track (bicycle) vehicle model showing the body-fixed coordinate frame, steering angle, and geometric parameters.

pointing upwards to form a right-handed coordinate system. Body-frame longitudinal and lateral velocities are denoted by $v_x$ and $v_y$, respectively, while the vehicle speed is defined as $v = \sqrt{v_x^2 + v_y^2}$.

A frequently used derived quantity is the side-slip angle $\beta$, defined as the angle between the vehicle's speed vector and its longitudinal axis,

$$\beta = \arctan 2(v_y, v_x). \tag{2-1}$$

Lowercase coordinates $(x, y)$ are used consistently to denote inertial-frame positions for compactness, rather than adopting separate uppercase notation often found in the literature. The employed coordinate frames, angles, and sign conventions are illustrated in Figure 2-1.

### 2-1-2 Kinematic Bicycle Model

The kinematic bicycle model neglects lateral tire slip and rotational dynamics, implying that the vehicle instantaneously follows the path dictated by its steering angle and forward velocity. First introduced by Riekert and Schunck in 1940 [41], this "simple" bicycle model remains widely used today [20]. It is called the kinematic bicycle model because the motion of the vehicle is described purely by geometric relationships. This leads to simple equations of motion that can be used in real-time for lower-speed applications such as parking and small-scale navigation [37]. A standard form of the kinematic bicycle model typically describes the rear wheel position as:

$$\dot{x}_r = v_x \cos(\psi), \tag{2-2}$$

$$\dot{y}_r = v_x \sin(\psi), \tag{2-3}$$

$$\dot{\psi} = \frac{v_x}{l} \tan(\delta), \tag{2-4}$$

where $(x_r,\ y_r)$ is the rear wheel's longitudinal and lateral position respectively, and $l$ is the wheelbase (distance between the front and rear wheels). The changes in longitudinal velocity can be modeled by a simple dynamic model that assumes perfect alignment of motor and friction forces with the vehicle's longitudinal axis:

$$\dot{v}_x = \frac{1}{m}(F_m + F_f), \tag{2-5}$$

where $m$ is the vehicle mass, $F_m$ is the motor force, and $F_f$ is the friction force caused by rolling resistance.

### 2-1-3 Dynamic Bicycle Model

Dynamic bicycle models extend the kinematic formulation by relaxing the assumption that the vehicle instantaneously follows the commanded path. By including inertia and tire slip dynamics, they capture transient effects that dominate at higher speeds or during aggressive maneuvers. Simple dynamic models perform well at moderate highway speeds, but caution is needed at very low or extremely high speeds [47]. The term "dynamic bicycle model" is used inconsistently in literature. In this thesis, "dynamic bicycle model" refers to a planar single-track model that includes lateral and longitudinal dynamics and yaw inertia, while neglecting roll, pitch, and suspension dynamics. Lateral tire forces are modeled as functions of tire slip angles. These slip angles represent the angular difference between the wheel heading and the direction of motion at each axle, and constitute the mechanism through which lateral forces are generated.

The corresponding equations of motion are:

$$\dot{x} = v_x \cos(\psi) - v_y \sin(\psi), \tag{2-6}$$

$$\dot{y} = v_x \sin(\psi) + v_y \cos(\psi), \tag{2-7}$$

$$\dot{\psi} = \omega, \tag{2-8}$$

$$\dot{v}_x = \frac{1}{m}(\hat{F}_{x,r} + \hat{F}_{x,f}\cos(\delta) - \hat{F}_{y,f}\sin(\delta)) + \omega v_y, \tag{2-9}$$

$$\dot{v}_y = \frac{1}{m}(\hat{F}_{y,r} + \hat{F}_{y,f}\cos(\delta) + \hat{F}_{x,f}\sin(\delta)) - \omega v_x, \tag{2-10}$$

$$\dot{\omega} = \frac{1}{I_z}\Big(l_f(\hat{F}_{y,f}\cos(\delta) + \hat{F}_{x,f}\sin(\delta)) - l_r\hat{F}_{y,r}\Big), \tag{2-11}$$

where $x, y$ denote the vehicle's position in the inertial frame, $l_f, l_r$ are the longitudinal distances from the center of gravity to the respective axles, $I_z$ is the rotational inertia about the vertical axis, and $\hat{F}_{x,i}$, $\hat{F}_{y,i}$ denote the longitudinal and lateral tire forces at axle $i \in \{f, r\}$. These forces are defined in each tire frames. The steering angle $\delta$ rotates the front-axle forces into the body frame, which is reflected in the $\sin(\delta)$ and $\cos(\delta)$ terms.

The first two equations describe the transformation of body-frame velocities into global coordinates. The third equation defines the yaw rate, while the fourth and fifth equations govern the longitudinal and lateral accelerations of the vehicle's center of gravity. These depend on the combined effects of propulsion, braking, and tire forces as well as the coupling between lateral and yaw motion. Finally, the last equation expresses the rotational dynamics, linking the yaw moment to the difference in lateral forces between the front and rear axles.

In this dynamical model, the tire forces $\hat{F}_x$ and $\hat{F}_y$ are typically represented using empirical tire models, such as the Pacejka "Magic Formula" [36]. This formula captures the nonlinear relationship between slip angle and lateral force, including force saturation, where beyond a certain slip angle the lateral tire force reaches a maximum and cannot increase further. Representing this saturation is important for modeling the physical limits of tire–road interaction at high lateral accelerations and is therefore standard in control-oriented prediction models for high-speed path tracking. The system identification process and the final identified model used in the MPPI control algorithm are discussed in Section 3-2-1.

## 2-2 Model Predictive Control

Model Predictive Control (MPC), also referred to as Receding-Horizon Control, is a powerful control strategy that anticipates future system behavior while explicitly accounting for constraints. Although MPC is not the primary contribution, it establishes the receding-horizon optimal control principle that MPPI approximates through stochastic sampling. Unlike conventional state-feedback controllers, MPC repeatedly solves an optimization problem at each time step to determine optimal control inputs for Multiple-Input Multiple-Output (MIMO) systems. This allows for smooth and adaptive control, making MPC a well-recognized method in path tracking for AVs [30]. Its key advantages include the ability to handle multiple variables and constraints in a unified framework, to incorporate different actuator models, and to maintain high performance under challenging conditions [48].

The main limitation of MPC is computational: solving constrained optimization problems online becomes challenging as model fidelity, horizon length, and constraint complexity increase [48, 46]. In addition, complex scenarios like hard constraints or aggressive maneuvers can render the solvers infeasible [46].

At each discrete time step, MPC formulates and solves a finite-horizon Optimal Control Problem (OCP) in an open-loop manner. The computed control sequence assumes perfect model knowledge and disturbance-free evolution over the prediction horizon. The first input of the resulting optimal control sequence is then applied to the system, after which the horizon "recedes" one step and the process repeats [40]. This receding-horizon principle combines the anticipatory nature of open-loop control with the robustness of closed-loop feedback.

To illustrate this more formally, assume the system follows a discrete-time model

$$x(k+1) = f(x(k), u(k)), \tag{2-12}$$

where $x(k)$ is the state at time $k$ and $u(k)$ is the control input. Over a prediction horizon of length $N$, the OCP can be written as

$$\{u^*(k), \cdots, u^*(k+N-1)\} = \arg\min_{\{u(k),\cdots,u(k+N-1)\}} \sum_{i=k}^{k+N-1} L\big(x(i), u(i)\big) + \phi\big(x(k+N)\big), \tag{2-13}$$

subject to constraints on states and inputs

$$x(k+1) = f(x(k), u(k)), \quad x(i) \in X, \quad u(i) \in U_s, \quad x(k+N) \in X_f. \tag{2-14}$$

Here, $L(\cdot)$ represents the running cost, typically combining tracking errors with penalties on control magnitude or rate, $\phi(\cdot)$ represents the terminal cost, encouraging stable or optimal end-point behavior. The set $X$, defines the state space, encoding constraints that restrict the system. The set $U_s$ defines the control space, encoding constraints like actuator limitations and rate bounds. Finally, $X_f$ denotes a terminal constraint set that restricts the predicted terminal state to a region associated with feasibility or closed-loop stability [34]. Many practical MPC formulations adopt quadratic weighted cost functions so that the resulting optimization problem takes the form of a Quadratic Program (QP) that can be solved efficiently using mature numerical optimization methods [40]. Tuning the weights associated with deviations of certain states or inputs from their reference values allows for prioritization of different objectives, for example penalizing abrupt steering more heavily than small tracking errors. After solving Eq. (2-13), only the first control input $u^*(k)$ is applied. Then the horizon shifts one step forward, and the procedure is repeated at the next time instant. This procedure is illustrated in Figure 2-2.

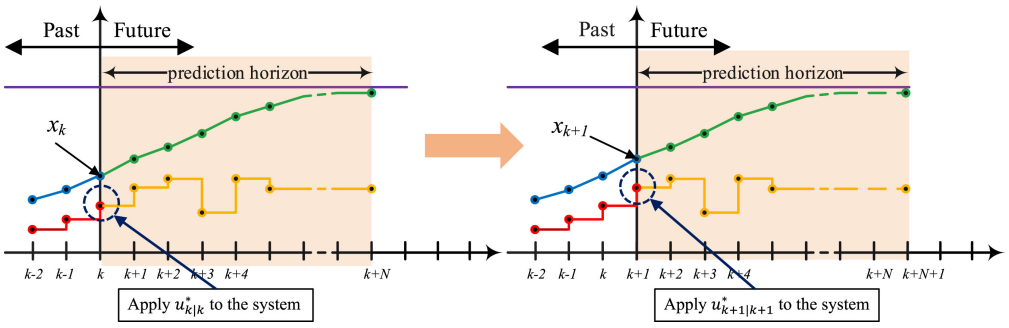



**Figure 2-2:** Schematic illustration of MPC. Past state and input trajectories are shown in blue and red, while predicted future states and control inputs are shown in green and yellow [45].

A key distinction between MPC and classical optimal control methods lies in how optimality is computed in practice. In dynamic programming, optimal control laws for both finite- and infinite-horizon problems are characterized by the Hamilton-Jacobi-Bellman (HJB) equation, which defines a value function over the continuous state space. Solving this equation yields the globally optimal feedback law, but computing the value function becomes intractable for nonlinear or high-dimensional systems due to the curse of dimensionality (Figure 2-3)[21]. MPC avoids explicit value-function computation by repeatedly solving a finite-horizon trajectory optimization from the current state and applying only the first control input, thereby constructing a receding-horizon feedback policy without solving the HJB equation directly. While this trade-off sacrifices global optimality, it enables real-time control for complex systems and motivates sampling-based approximations such as MPPI, which further relax the need for explicit optimization.

The formulations given above apply in theory to a wide variety of discrete-time models. However, as discussed in the previous section, many state-of-the-art vehicle models contain nonlinear terms, which often lead to non-convex optimization problems for control, making real-time solving potentially time-consuming. To mitigate this, several methods have been proposed to recast or approximate a wide variety of problems as a linear QP problem. This ranges from Linear Time-Invariant (LTI) MPC, which relies on a linear model from the outset

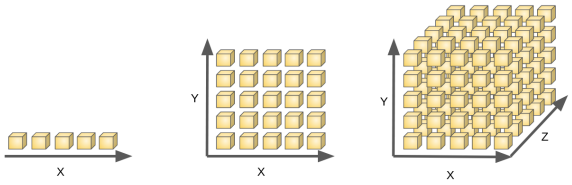



**Figure 2-3:** Illustration of the curse of dimensionality: the number of grid elements increases exponentially with each additional state dimension [35].

or to linearize the nonlinear system around a fixed operating point,

$$x(k+1) = Ax(k) + Bu(k), \tag{2-15}$$

to more adaptive approaches such as Linear Time-Varying (LTV) and Linear Parameter-Varying (LPV) MPC [60, 13].

### 2-2-1 Advanced Model Predictive Control

To address the limitations of classical linear MPC, various extensions have been developed. Nonlinear MPC (NMPC) extends the MPC framework by incorporating a nonlinear prediction model, rather than a linear or parameter-varying approximation, directly into the optimization problem. This makes it possible to fully capture vehicle and tire nonlinearities [19]. While this enhances model accuracy, it also introduces non-convexity into the optimization problem, substantially increasing computational effort. Commonly used toolboxes for solving NMPC include ACADOS [53], FORCES Pro [12], YALMIP [32], and CasADi [5], but the challenge of reliably solving a non-convex problem within strict time constraints persists [48].

Several studies have shown that incorporating nonlinear vehicle models within MPC frameworks yields measurable performance gains in aggressive or high-speed driving scenarios [15, 9]. In particular, relaxing low-speed or planar-motion assumptions improves tracking accuracy and enables real-time trajectory tracking at higher speeds.

A specialized MPC formulation tailored for path following is Model Predictive Contouring Control (MPCC). MPCC focuses on tracking or "contouring" a desired path with minimal deviation and maximum progress along the path, making it particularly relevant for path-tracking applications. Unlike traditional model predictive control approaches, MPCC explicitly incorporates a contouring error into its optimization problem, enabling a trade-off between accuracy and progress. Lam et al. [28] introduced MPCC by explicitly separating tracking error into contouring error (normal deviation from the path) and lag (deviation along the path), as $\epsilon^c$ and $\epsilon^l$ respectively. Figure 2-4 shows a graphical interpretation including the approximations of $\epsilon^c$ and $\epsilon^l$ for the next timestep. Using the MPCC framework, traveled distance on the reference path within the prediction horizon can be maximized, making it especially promising for autonomous racing, where the lowest lap times have to be reached [31, 27].

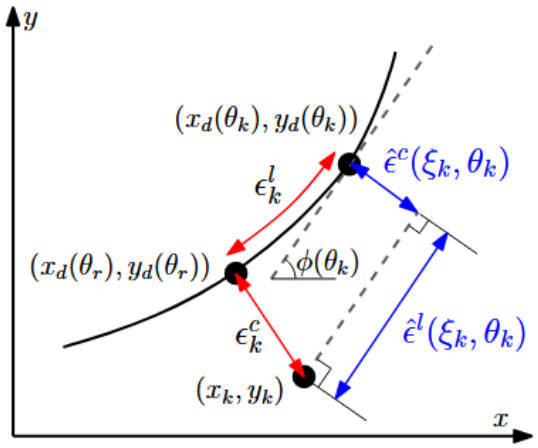



**Figure 2-4:** Geometric interpretation of contouring error $\epsilon^c$ and lag error $\epsilon^l$ in MPCC [28].

## 2-3 Model Predictive Path Integral

MPC provides a principled framework for optimizing control inputs over a finite horizon under physical constraints, but solving large nonlinear programs at high update rates becomes challenging for nonlinear dynamics and nonconvex costs. Model Predictive Path Integral (MPPI) retains the receding-horizon structure of MPC while replacing deterministic nonlinear programming with a sampling-based approximation of the stochastic optimal control problem. This sampling-based reformulation enables massively parallel rollout evaluations (typically on GPUs), making real-time control feasible for high-dimensional nonlinear systems [23].

Because key results in stochastic optimal control are most naturally derived in continuous time, a continuous-time formulation is introduced and then discretized for implementation.

The models introduced above are deterministic. However, in practice, unmodelled dynamics and disturbances affect state evolution. A common representation is the Stochastic differential equations (SDEs), of the form:

$$dx = f(x_t, u_t, t)dt + B(x_t, u_t, t)dw, \tag{2-16}$$

where $dw$ represents increments of Brownian motion (or a Wiener process), and $B(\cdot)$ represents how stochastic noise enters the system (diffusion). Conceptually, SDEs capture how the system follows the deterministic trend given by $f(\cdot)$ but continuously fluctuates around this trend due to inherent randomness, defined by $B(\cdot)$. The objective is to find a control sequence that minimizes the expected cost

$$\mathbb{E}\Big[\phi(x_T) + \int_0^T L(x_t, u_t, t)dt\Big], \tag{2-17}$$

where $\phi(x_T)$ denotes the terminal cost and $L(x_t, u_t, t)$ the running cost, and $T$ is the continuous-time prediction horizon. In this formulation, the objective is defined as an expectation over stochastic system trajectories. By forming control updates as cost-weighted averages over these sampled trajectories, MPPI replaces deterministic nonlinear optimization with an approximation of the expected cost through Monte Carlo sampling. This trajectory sampling

allows stochastic effects and modeling errors to be reflected implicitly through the distribution of rollouts [23]. An example set of rolled out trajectories and the resulting control sequence are illustrated in Figure 2-5.

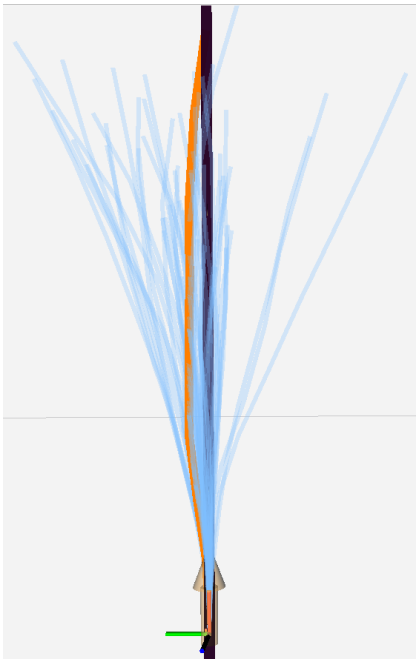


**Figure 2-5:** Illustration of MPPI rollouts: sampled control trajectories (blue) and the resulting updated control sequence (orange).

MPPI can be derived from two complementary viewpoints. First, from the Path Integral (PI) viewpoint, it recasts the stochastic optimal control problem in terms of forward simulations of randomly sampled control inputs resulting in "noisy" trajectories, each of which is weighted by the exponential of its total cost. Second, from an information-theoretic viewpoint, MPPI identifies a trajectory distribution that balances minimizing expected cost with remaining close to a nominal or uncontrolled distribution. Conceptually, the expected cost reflects tracking error or other control objectives, while the cost of deviating from the nominal distribution captures the penalty on control effort. The following subsections present both derivations.

### 2-3-1 Path Integral Formulation

The PI formulation provides the theoretical foundation of MPPI by linking stochastic optimal control to a forward-sampling interpretation of the HJB equation. Solving this problem directly via the stochastic HJB equation is typically infeasible, because most numerical solution methods exhibit computational complexity that increases exponentially with the dimensionality of the state space.

The associated stochastic HJB Partial Differential Equation (PDE) is nonlinear and typically intractable in high-dimensional settings. The objective is to exploit the Feynman–Kac formula, which provides a probabilistic representation for solutions of linear second-order PDEs. To enable such a representation, the nonlinear HJB equation must first be transformed into a linear form.

This linearization is possible under specific structural assumptions. First, the system dynamics are assumed to be control-affine and the control cost quadratic. Second, the control–noise matching condition $BB^\top = \lambda G R^{-1} G^\top$ must hold. This condition links the diffusion matrix $B(x,t)$, control matrix $G(x,t)$, control cost matrix $R$, and temperature parameter $\lambda$. Intuitively this condition states that, directions of high uncertainty must be directly actuated and not excessively penalized. Under these assumptions, the nonlinear HJB equation for the value function $V(x,t)$ can be rendered linear through the logarithmic transformation $V(x,t) = -\lambda \log \Psi(x,t)$, which defines the function $\Psi(x,t)$, referred to as the desirability function [50, 59]. The scalar $\lambda$ appears both in the matching condition and in the transformation, and scales the sensitivity of the optimal policy to cost. The resulting linear PDE admits a forward-in-time probabilistic representation of $\Psi$ via the Feynman–Kac formula

$$\Psi(x) = \mathbb{E}_{\mathbb{P}}\Big[\exp\Big(-\frac{1}{\lambda}\Big(\phi(x_T) + \int_0^T L(x_t,t)dt\Big)\Big)\Big], \tag{2-18}$$

where $\mathbb{P}$ denotes the distribution of trajectories under the uncontrolled dynamics $dx = f(x,0,t)dt + B(x,0,t)dw$. In this representation, the expectation under $\mathbb{P}$ integrates over all possible stochastic trajectories starting from state $x$, and the exponential term inside the expectation acts as a weighting factor. Trajectories that accumulate a smaller cost contribute more to $\Psi(x)$, while those with higher cost are exponentially suppressed. In continuous time, the optimal control can be expressed in terms of the gradient of the value function, which, under the logarithmic transformation, becomes a weighted expectation over stochastic trajectories.

In practice, this expectation cannot be evaluated analytically and is instead approximated numerically via Monte Carlo sampling. The expectation is taken with respect to the uncontrolled system dynamics, which is theoretically sound but often yields poor sampling efficiency because low-cost trajectories are unlikely under this distribution [56]. In discrete time, this approximation leads to a cost-weighted average over sampled control perturbations:

$$u^*(k) \approx u(k) + \sum_{j=1}^{J} \frac{\exp\big(-\frac{1}{\lambda}S_j\big)}{\sum_{i=1}^{J}\exp\big(-\frac{1}{\lambda}S_i\big)}\delta u_j(k), \tag{2-19}$$

where

$$S_j = \phi(x(N)) + \sum_{k=0}^{N-1} L\big(x_j(k), u_j(k)\big) \tag{2-20}$$

denotes the cost-to-go of the $j$-th sampled trajectory and $\delta u_j(k)$ represents the corresponding control perturbations. Eq. (2-19) is the core MPPI update: each control sample is weighted by its trajectory cost, and the nominal control sequence is shifted towards low-cost regions of the control space.

### 2-3-2 Information-Theoretic Perspective

The path-integral derivation assumes, among other, control-affine dynamics, meaning the control enters linearly in the dynamics through the term $G(x,t)u_t$. While this assumption holds for many robotic and vehicle systems, it limits the generality of the approach.

To address this limit, the same update law can also be derived using an information-theoretic framework, which frames control as inference over trajectory distributions and is often presented as more broadly applicable than the PI linearization route [58]. Let $S(\tau)$ denote a cost-to-go function for a trajectory of duration $N$, and let $\tau$ be a random variable representing the full trajectory (or all variables that define it) starting from an initial condition $x_0$, drawn from a probability measure $\mathbb{P}$ representing the uncontrolled system. With the temperature parameter $\lambda > 0$, the free energy function is defined as

$$F(S, \mathbb{P}, x_0, \lambda) = -\lambda \log\Big(\mathbb{E}_{\mathbb{P}}\Big[\exp\Big(-\frac{1}{\lambda}S(\tau)\Big)\Big]\Big). \tag{2-21}$$

This function comes from statistical thermodynamics, where the Helmholtz free energy has the same logarithmic-exponential structure. In thermodynamics, this quantity represents the maximum amount of useful work that can be extracted from a system at constant temperature and volume. However the quantity of the free energy differs in meaning when describing a control system. Here it reflects a trade-off between minimizing trajectory cost and maintaining exploration, with $\lambda$ controlling how strongly low-cost trajectories are favored. By applying Jensen's inequality, one obtains the following lower bound on the free energy:

$$F(S, \mathbb{P}, x_0, \lambda) \leq \mathbb{E}_{\mathbb{Q}}\big[S(\tau)\big] + \lambda \mathbb{KL}(\mathbb{Q}\|\mathbb{P}), \tag{2-22}$$

where $\mathbb{Q}$ is a controlled trajectory distribution and $\mathbb{KL}(\mathbb{Q}\|\mathbb{P})$ denotes the Kullback-Leibler (KL) divergence between $\mathbb{Q}$ and $\mathbb{P}$ [58]. The KL divergence measures the discrepancy between distributions via the Radon–Nikodym derivative. In this definition, $\mathbb{E}_{\mathbb{Q}}[S(\tau)]$ is the expected cost-to-go under $\mathbb{Q}$, while $\mathbb{KL}(\mathbb{Q}\|\mathbb{P})$ penalizes how much $\mathbb{Q}$ deviates from $\mathbb{P}$. Minimizing the right-hand side of Eq. (2-22) yields the optimal distribution $\mathbb{Q}^*$ that balances two objectives: minimizing the expected cost and keeping the controlled distribution close to the uncontrolled one. The distribution that achieves equality is

$$\frac{d\mathbb{Q}^*}{d\mathbb{P}} = \frac{\exp(-S(\tau)/\lambda)}{\mathbb{E}_{\mathbb{P}}\big[\exp(-S(\tau)/\lambda)\big]}. \tag{2-23}$$

which assigns higher probability to low-cost trajectories. Sampling from $\mathbb{Q}^*$ would therefore yield trajectories whose expected cost is minimal among all possible distributions.

In practice, $\mathbb{Q}^*$ cannot be sampled directly. Instead, MPPI represents the current policy by a simpler parametric distribution $\mathbb{Q}_{U,\Sigma}$ (representing a particular controlled distribution) with mean control sequence $U = \{u(0), \cdots, u(N-1)\}$ and covariance $\Sigma$, and iteratively pushes this distribution toward $\mathbb{Q}^*$ (Figure 2-6) by solving

$$U^* = \arg\min_{U} \mathrm{KL}(\mathbb{Q}^*\|\mathbb{Q}_{U,\Sigma}), \tag{2-24}$$

where $U^* = \{u^*(0), \cdots, u^*(N-1)\}$ is the computed control sequence. This objective explicitly minimizes the divergence between the current control distribution and the optimal one that concentrates probability mass around low-cost trajectories. The minimization produces the same weighting rule as in Eq. (2-19), because $\mathbb{Q}^*$'s density is proportional to $\exp(-S/\lambda)$. In practice, MPPI approximates $\mathbb{Q}^*$ through sampling: by generating a large number of rollouts under the nominal control, computing their costs, and updating the control mean toward the weighted average implied by this distribution. Sampling is performed around a "nominal"

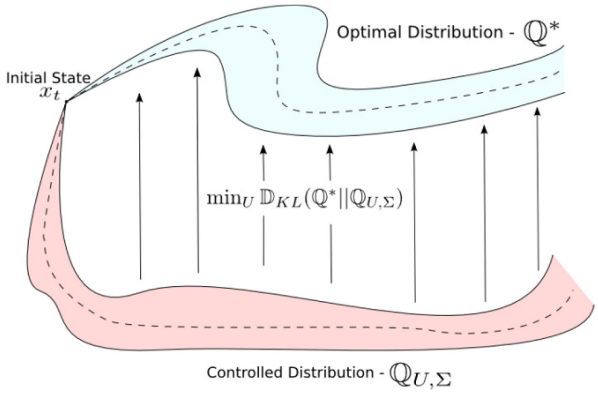



**Figure 2-6:** Information-theoretic interpretation of MPPI: the controlled trajectory distribution is iteratively shifted toward the optimal distribution via KL divergence minimization [58].

control sequence, in practice at each timestep the previous computed control trajectory $U^*$ is taken as the new nominal control sequence, resulting in:

$$\delta u_j(k) \sim \mathcal{N}(0, \Sigma), \qquad u_j(k) = u^*(k) + \delta u_j(k), \tag{2-25}$$

which generate a set of trajectories $\tau_j$ through forward simulation.

In summary, both the path integral and information-theoretic viewpoints lead to the same practical outcome. The information-theoretic formulation is more general because it does not require the linear control-affine structure or the control–noise matching used by PI to linearize the HJB. Instead, it operates directly on probability measures over trajectories and only requires absolute continuity of the controlled trajectory distribution $\mathbb{Q}$ with respect to the uncontrolled distribution $\mathbb{P}$. This condition ensures that trajectories emphasized by the controller are already present in the sampling distribution, allowing their probabilities to be reweighted through a well-defined likelihood ratio.

### 2-3-3 MPPI Algorithm

The derivations in the previous sections motivate MPPI's exponential reweighting rule. In a discrete-time receding-horizon implementation, several practical details are commonly introduced to improve numerical conditioning and closed-loop behavior. First, the rollout cost-to-go is often temporally discounted using the factor $\gamma$, which shifts the emphasis away from the late-horizon cost contributions and improves robustness to prediction errors. The discounted cost-to-go for rollout $j$ then becomes

$$S_j = \phi(x(N)) + \sum_{k=0}^{N-1} \gamma^k \, L(x_j(k), u_j(k)). \tag{2-26}$$

In contrast to constrained MPC, in MPPI constraint handling is embedded in the rollout evaluation. Input bounds are enforced by saturating (or squashing) sampled controls before forward simulation, ensuring that each rollout respects admissible actuator limits. State constraints (e.g., track boundaries or obstacle avoidance) are encoded as soft penalties by

augmenting the running cost with a constraint term $C(\cdot)$. Under the importance-weighted update in Eq. (2-19), trajectories that violate constraints incur large costs and therefore receive negligible importance weights, biasing the update toward constraint-compliant rollouts. The resulting final cost-to-go becomes

$$S_j = \phi(x(N)) + \sum_{k=0}^{N-1} \gamma^k \Big( L(x_j(k), u_j(k)) + C(x_j(k)) \Big). \tag{2-27}$$

For numerical stability, the importance weights are computed using a shifted cost to prevent overflow:

$$\rho = \min_j S_j, \qquad \eta = \sum_{j=1}^{J} \exp\Big(-\tfrac{1}{\lambda}(S_j - \rho)\Big), \qquad w_j = \frac{1}{\eta} \exp\Big(-\tfrac{1}{\lambda}(S_j - \rho)\Big), \tag{2-28}$$

where $\rho$ is the minimum cost across all sampled trajectories, $\eta$ is a normalization constant that ensures the weights sum to one, and $w_j$ is the resulting normalized importance weight of trajectory $j$. This normalization preserves the relative weighting implied by the exponential cost while improving numerical conditioning. The update in Eq. (2-19) then uses $w_j$ instead of the raw exponential weights, yielding the same control update with improved numerical stability.

Throughout this thesis, MPPI refers to the baseline discrete-time, receding-horizon implementation of path integral control. As a practical post-processing step, many MPPI implementations smooth the updated control sequence over the horizon. Williams et al. apply a Savitzky–Golay filter independently to each control dimension [58, 43]. This smoothing reduces high-frequency variation but is heuristic and should be interpreted as actuator-friendly post-processing rather than part of the MPPI optimality conditions. A consice version of the MMPPI algorithm is given in Algorithm 1.

## 2-4 MPPI Extension Concepts

Practical MPPI implementations frequently modify the sampling process or control parametrization to improve numerical conditioning and better accommodate real-time control constraints. The goal of this section is to define the concepts and notation used later. The experimental motivation and empirical evaluation of these variants are presented in Section 4-4.

### 2-4-1 Dynamic Covariance MPPI

Standard MPPI employs a fixed sampling covariance and temperature parameter to generate stochastic control perturbations. Together, these parameters determine the scale of exploration and the concentration of importance weights in the control update. While fixed choices are simple to implement, they implicitly assume that the same exploration–exploitation tradeoff is appropriate across all operating conditions, which is rarely the case in practice.

To address this limitation, several adaptive variants of MPPI have been proposed that modify the sampling distribution online [62, 63]. A common principle underlying these approaches is

**Algorithm 1** MPPI algorithm.

**Given:** horizon $N$, number of samples $J$, temperature $\lambda$
**Initialize:**
  Nominal control sequence $U^* = \{u_0, \cdots, u_{N-1}\}$, Sampling covariance $\Sigma$, Temperature $\lambda$
  Running cost $L(\cdot)$ with reference state $\bar{x}_{\text{ref}}$, and weights $q$
**while** control is active **do**
  Observe current state $x_0$
  **Sample trajectories**
  **for** $j = 1, \cdots, J$ **do**
    Sample noise sequence $\delta u_j \sim \mathcal{N}(0, \Sigma)$
    Form perturbed controls $u_k^{(j)} = u_k^* + \delta u_{j,k}$
    Roll out dynamics and compute cost-to-go $S_j$ (Eq. (2-20))
  **end for**
  $\rho \leftarrow \min_j S_j$
  $\eta \leftarrow \sum_{j=1}^{J} \exp\Big(-\frac{1}{\lambda}(S_j - \rho)\Big)$
  **for** $j = 1, \cdots, J$ **do**
    $w_j \leftarrow \frac{1}{\eta} \exp\Big(-\frac{1}{\lambda}(S_j - \rho)\Big)$
  **end for**
  **Update control sequence**
  **for** $k = 0, \cdots, N-1$ **do**
    $u_k^* \leftarrow u_k^* + \sum_{j=1}^{J} w_j \delta u_{j,k}$
  **end for**
  Optionally apply temporal smoothing to $U$
  Apply control input $u_{\text{applied}} = u_0^*$
  Shift horizon $U^* \leftarrow \{u_1^*, \cdots, u_{N-1}^*, u_{\text{init}}\}$
**end while**

to adapt the sampling covariance based on the distribution of trajectory costs, with the aim of balancing exploration and convergence more effectively.

In MPPI, the sampling covariance $\sigma_i$ determines the scale of exploration in control space for control input $i$. A fixed covariance therefore implicitly assumes that all control dimensions are equally sensitive to perturbations over time. The convergence rate of MPPI depends strongly on the choice of the sampling covariance, and commonly used fixed and isotropic covariance matrices are generally suboptimal [62].

To reduce sensitivity to manual covariance tuning, an online adaptation of a diagonal sampling covariance can be employed. Let $\delta u_j(k) \in \mathbb{R}^\nu$ denote the control perturbation applied at time $k$ in rollout $j$ with the control dimension denoted by $\nu$, and let $w_j$ be the corresponding importance weight. For each control dimension, the magnitude of perturbations that contribute to successful rollouts provides a direct indication of how much exploration is useful in that direction.

This intuition is implemented by forming a weighted average of the squared perturbations, averaged over the horizon and across rollouts. Denoting the diagonal of $\Sigma$ by $c \in \mathbb{R}^\nu$, the

covariance is updated using an exponential moving average,

$$c_{\text{new}} = (1 - \alpha_\Sigma)\, c_{\text{old}} + \alpha_\Sigma \underbrace{\frac{1}{N} \sum_{k=0}^{N-1} \sum_{j=1}^{J} w_j \left(\delta u_j(k) \odot \delta u_j(k)\right)}_{\text{weighted average of squared perturbations}}, \tag{2-29}$$

where $\alpha_\Sigma \in (0, 1)$ controls the adaptation rate, $\odot$ denotes element-wise multiplication. To prevent premature collapse of exploration, a small positive floor $\kappa > 0$ is added,

$$c \leftarrow c_{\text{new}} + \kappa, \tag{2-30}$$

ensuring continued stochastic excitation even when the control distribution becomes highly concentrated. This covariance adaptation rule follows the open-source implementation accompanying [38, 51].

In summary, perturbations that consistently contribute to low-cost trajectories increase the estimated covariance, while ineffective perturbations lead to its reduction. Intuitively, this update increases exploration in control directions where larger perturbations consistently yield low-cost trajectories, and reduces it in directions where only small deviations are beneficial.

The adaptive covariance can therefore be interpreted as a task-dependent control-space trust region that evolves with operating conditions while affecting only the scale of exploration. The cost-weighted mean update remains unbiased, thereby preserving the fundamental optimality structure of MPPI.

### 2-4-2 Smooth MPPI

Smooth MPPI (SMPPI) addresses abrupt control behavior by reformulating the sampling space rather than modifying the resulting control sequence [25]. Instead of sampling control actions directly, SMPPI samples higher-order control variables, typically control rates, which are integrated over the prediction horizon to obtain the actual control inputs.

This parameterization induces temporal correlation in the generated control sequence: large instantaneous changes in the control signal can only arise from sustained rate perturbations across multiple timesteps. Consequently, high-frequency control variations are structurally suppressed, and smoothness emerges from the integration dynamics rather than from post-processing or heuristic filtering.

In practice, the magnitude of rate variations is governed by the sampling covariance in the rate space. Reducing this covariance limits the amplitude of control-rate perturbations and therefore bounds the rate of change of the integrated control signal. Unlike standard MPPI, where covariance reduction directly restricts the amplitude of the control input itself, covariance reduction in SMPPI constrains the derivative of the control signal while still allowing large control magnitudes to be accumulated over time. This distinction enables smoother actuator commands without sacrificing the ability to generate aggressive steering inputs when required [25].

Sampling in the space of action derivatives allows variance to be reduced in the higher-order space without restricting the achievable control magnitude over the prediction horizon. Rapid

actuator fluctuations are thereby discouraged, mitigating mechanical stress and reducing actuator wear. At the same time, the integration of control-rate perturbations across the horizon enables coherent adjustments to the control sequence, which can improve performance in dynamically changing conditions.

### 2-4-3 Low-pass Filtered Sampling MPPI

An alternative approach to improving control smoothness in MPPI is to modify the temporal structure of the sampling noise rather than the control representation itself. In standard MPPI, control perturbations are sampled as time-uncorrelated Gaussian noise, which introduces significant high-frequency content into sampled control trajectories [54]. While such exploration is mathematically valid, it often leads to unnecessarily rapid variations in computed control signals.

Introducing temporal correlation in the sampling process constrains the perturbations to smoother trajectories that are more consistent with physical actuator dynamics. This correlation can be achieved either by sampling directly in the frequency domain [54] or by filtering time-domain noise through a low-pass filter prior to rollout evaluation [24].

A simple but effective low-pass filter is adopted. Rather than sampling directly in the frequency domain, temporal correlation is introduced by applying a causal low-pass filter along the horizon to each sampled control sequence prior to rollout. Specifically, for each rollout $j$,

$$\tilde{v}_j(0) = v_j(0), \qquad \tilde{v}_j(k) = \alpha_f \, \tilde{v}_j(k-1) + (1-\alpha_f)\, v_j(k), \quad k = 1, \dots, N-1, \tag{2-31}$$

with smoothing factor $\alpha_f \in (0, 1)$. Using the filter for smoothing of the samples is called Low-pass Filtered Sampling MPPI (LFS-MPPI). Although less expressive than frequency-domain sampling methods, this approach is computationally lightweight, integrates naturally with the receding-horizon structure of MPPI, and was found to be sufficient for improving control smoothness in the experimental scenarios considered. Importantly, the core MPPI update rule remains unchanged, and no external filtering is applied to the final control signal.

# Chapter 3

# Methodology

This chapter presents the concrete control pipeline used for real-time implementation of the baseline Model Predictive Path Integral (MPPI) controller, together with the experimental framework that is shared across simulation and laboratory evaluations. The objective is to establish a consistent methodological foundation for all subsequent performance comparisons.

The controller design is guided by practical constraints, including real-time execution requirements, software integration, and the need to deploy identical control logic in both simulation and hardware. Within this setting, three aspects are addressed. First, the cost function and reference generation framework are introduced. Second, the baseline MPPI controller is validated, with particular attention to systematic effects that may arise even under nominal model matching. Third, the test environments, reference trajectories, and experimental parameters used throughout Chapters 4 and 5 are described.

## 3-1 Control Architecture

While Chapter 2 introduced MPPI at the conceptual level, the present section specifies the concrete real-time control architecture used in this thesis. It details how the cost function, state information, prediction, and reference generation, are composed into a closed-loop pipeline.

### 3-1-1 Cost Function Formulation

The objective function used within the MPPI planner defines the optimization criterion for all sampled trajectories and therefore has a direct influence on the resulting control behavior. The structure of the cost function is intentionally kept simple. The primary focus lies on analyzing the behavior of MPPI and its algorithmic extensions, rather than on designing or tuning a task-specific optimal objective. Maintaining a fixed cost formulation ensures that differences observed between controller variants can be attributed to algorithmic changes rather than to variations in the objective itself.

The running cost $L(\cdot)$ is evaluated at each prediction step and penalizes deviations from a locally defined reference trajectory, as well as selected state and control quantities. Let $(x, y, \psi)$ denote the vehicle position and yaw angle, $(v_x, v_y)$ the longitudinal and lateral velocities, $\omega$ the yaw rate, and $u = [u_{\text{thr}}, u_{\text{str}}]^\top$ the throttle and steering inputs. The state vector is defined as $\bar{x} = \begin{bmatrix} x & y & \psi & v_x & v_y & \omega \end{bmatrix}^\top$. The reference trajectory provides, for each timestep over the prediction horizon $N$, the reference position $(x_{\text{ref}}, y_{\text{ref}})$, yaw angle $\psi_{\text{ref}}$, and reference speed $v_{\text{ref}}$. The running cost is defined as

$$\begin{aligned} L(\bar{x}, u) = q_{\text{pos}} \left( (x - x_{\text{ref}})^2 + (y - y_{\text{ref}})^2 \right) + q_{\text{head}} (\psi - \psi_{\text{ref}})^2 + q_v \left( \sqrt{v_x^2 + v_y^2} - v_{\text{ref}} \right)^2 \\ + q_{u_{\text{thr}}} u_{\text{thr}}^2 + q_{\Delta u_{\text{thr}}} (\Delta u_{\text{thr}})^2 + q_{u_{\text{str}}} u_{\text{str}}^2 + q_{\Delta u_{\text{str}}} (\Delta u_{\text{str}})^2 + c_{\text{lane}}(\bar{x}) + c_{\text{obst}}(\bar{x}), \end{aligned} \tag{3-1}$$

where $\Delta u$ denotes the change in control input between successive timesteps, the weights are noted as $q_i$ for each error respectively, and $c_{\text{lane}}(\bar{x}), c_{\text{obst}}(\bar{x})$ denote the costs associated with lane and obstacle constraints respectively.

A soft lane-keeping penalty is included to discourage trajectories that exceed a predefined margin within the lane boundaries. Let $w_{\text{lane}}$ denote the lane width and $m_{\text{lane}}$ a safety margin inside the lane. The lane cost is defined as

$$c_{\text{lane}} = \text{clip} \left( q_{\text{lane}} \Gamma \left( |\epsilon^l| - \left( \tfrac{1}{2} w_{\text{lane}} - m_{\text{lane}} \right) \right), 0, c_{\max} \right), \tag{3-2}$$

where $\epsilon^l$ denotes the lateral deviation from the lane centerline and $\Gamma(\cdot)$ is a is a smooth approximation of the hinge function $\max(0, e)$, implemented as a scaled softplus function with adjustable sharpness. The cost is saturated at $c_{\max}$ to prevent excessive dominance of the lane penalty in the MPPI importance weights, which could otherwise lead to numerical degeneracy.

For obstacle avoidance scenarios, an additional soft constraint penalty is incorporated into the running cost to discourage trajectories that approach or intersect obstacles. Obstacles are modeled with elliptical safety regions in the plane, enabling smooth, distance-based penalties that remain compatible with the sampling-based MPPI formulation.

Let $p = [x\ y]^\top$ denote the vehicle position and $c = [c_x\ c_y]^\top$ the obstacle center. For a static obstacle with fixed orientation, the implicit ellipse representation is defined as

$$\Phi_{\text{obs}}(x, y) = (p - c)^\top Q (p - c) - 1, \tag{3-3}$$

where $Q \in \mathbb{R}^{2\times 2}$ encodes the obstacle shape and orientation. The interior of the ellipse corresponds to $\Phi_{\text{obs}} < 0$, while $\Phi_{\text{obs}} = 0$ defines the obstacle boundary.

A smooth obstacle penalty is then constructed as

$$c_{\text{obs}} = \text{clip} \left( w_{\text{obs}} \Gamma \left( -\Phi_{\text{obs}}(x, y) + m_{\text{obs}} \right), 0, c_{\max} \right), \tag{3-4}$$

where $q_{\text{obs}}$ is the obstacle weight, $m_{\text{obs}}$ is a safety margin that inflates the effective obstacle size, and $\Gamma(\cdot)$ is the same softplus function used for the lane penalty. As with the lane cost, saturation at $c_{\max}$ prevents the obstacle term from dominating the MPPI weighting.

For dynamic obstacles, the obstacle center $c(k)$ and orientation $\theta_{\text{obs}}(k)$ are time-varying over the prediction horizon. In this case, the ellipse matrix is defined as

$$Q(k) = E(\theta_{\text{obs}}(k)) \begin{bmatrix} a^{-2} & 0 \\ 0 & b^{-2} \end{bmatrix} E^\top(\theta_{\text{obs}}(k)), \tag{3-5}$$

where $a$ and $b$ denote the ellipse semi-axes and $E(\theta)$ is the planar rotation matrix. The obstacle cost is evaluated at each prediction step using the corresponding $c(k)$ and $Q(k)$, with the obstacle center propagated along the reference trajectory at a constant speed.

Both the lane and obstacle penalties are weighted several orders of magnitude higher than nominal tracking terms. Furthermore, penetration of the obstacle safety envelope incurs a penalty that is an order of magnitude larger than the lane violation cost. This hierarchy explicitly biases the controller toward leaving the lane rather than risking a collision when a trade-off is unavoidable.

The numerical values of the cost weights used throughout the experiments are summarized in Table 3-1. Notably, no explicit penalty on steering rate is applied. This design choice ensures that high-frequency steering behavior, if present, emerges from the MPPI sampling and weighting mechanism itself rather than being directly suppressed through the objective. The rollout discount factor is set to $\gamma = 0.95$ in all experiments.

**Table 3-1:** Cost function weights used in the MPPI objective formulation.

| **Cost term** | **Weight** |
|---|---|
| Position error $q_{\text{pos}}$ | 1.0 |
| Heading error $q_{\text{head}}$ | 0.1 |
| Speed error $q_v$ | 0.1 |
| Throttle magnitude $q_{u_{\text{thr}}}$ | 0.01 |
| Throttle rate $q_{\Delta u_{\text{thr}}}$ | 0.01 |
| Steering magnitude $q_{u_{\text{str}}}$ | 0.01 |
| Steering rate $q_{\Delta u_{\text{str}}}$ | 0.0 |
| Lane Penalty $q_{\text{lane}}$ | 100.0 |
| Obstacle Penalty $q_{\text{obs}}$ | 1000.0 |

Overall, the cost function is deliberately formulated as a reference-tracking objective that penalizes deviations in position, heading, and speed relative to the desired trajectory. Higher-level notions such as control smoothness or oscillation suppression are intentionally not encoded directly in the objective, allowing their emergence, or lack thereof, to be attributed to the properties of the control algorithm itself.

### 3-1-2 Reference generation

At each control update, a local reference trajectory is generated from the global track definition and the current estimated vehicle pose. The closest point on the centerline is identified by projection, and the reference is extended forward along the centerline according to the reference speed and the controller timestep. This yields a prediction horizon-length sequence of desired positions and headings that is used by the running cost during MPPI rollouts. Anchoring the reference to the current state estimate at every update ensures that the controller remains well-defined under deviations from the centerline while maintaining consistency with the global track geometry. The same reference generation module is used across all simulation and laboratory experiments.

All reference trajectories used are defined entirely within a local coordinate frame taken from a (simulated) motion capture system. The reference paths exist only as virtual constructs within the controller and are not associated with any physical markers or environmental cues in the simulation or laboratory. As a result, trajectory tracking performance depends solely on the estimated vehicle state and the internally generated reference, rather than on external perception, localization, or mapping information.

### 3-1-3 State estimation

Because the motion capture system does not provide direct velocity measurements, linear and angular velocities are estimated from the measured position and orientation signals. To this end, a Finite-Difference (FD) scheme over a short sliding time window is employed. At each pose update, the most recent $P = 5$ position and yaw samples $(x, y, \psi)$ and their timestamps are stored in a fixed-length buffer. Linear and angular velocities are then approximated using an end-point finite difference across this window,

$$\dot{x} \approx \frac{x(k_P) - x(k_0)}{t(k_P) - t(k_0)}, \quad \dot{y} \approx \frac{y(k_P) - y(k_0)}{t(k_P) - t(k_0)}, \quad \dot{\psi} \approx \frac{\Delta\psi}{t(k_P) - t(k_0)}, \tag{3-6}$$

where the yaw difference $(\Delta\psi)$ is unwrapped to avoid discontinuities at $\pm\pi$. The resulting translational velocities are expressed in the body frame by rotating the world-frame estimates using the current measured yaw angle. This approach yields a low-latency speed estimate that is consistent with the controller state definition and avoids additional filtering-induced delay.

## 3-2 MPPI implementation

Several MPPI implementations are currently available, ranging from CPU-based Python libraries to GPU-accelerated frameworks. A PyTorch-based MPPI implementation is used [51], which follows the formulation introduced by Williams et al. [57]. The package is implemented entirely in Python and integrates naturally with the ROS-based software stack used in the experimental setup.

The MPPI implementation is structured as a generic planner class that takes as input a discrete-time prediction model, a running cost function, and a set of sampling and weighting parameters. At each control iteration, a batch of control sequences is sampled, rolled out through the prediction model, and evaluated using the specified cost function. A key implication of this design is that the cost function must be formulated in a fully online manner. At each control timestep, the objective must be evaluated based on the current estimated state and the reference information available at that instant. As a result, reference generation, cost evaluation, and control optimization are tightly coupled within the control loop, the software pipeline is visualized in Figure 3-1.

### 3-2-1 Prediction Model and Integration Scheme

MPPI evaluates candidate control sequences by rolling out a discrete-time prediction model over the planning horizon. In this thesis, either a kinematic bicycle model or a dynamic bicycle model is used for prediction. These are referred to hereafter as the kinematic model and

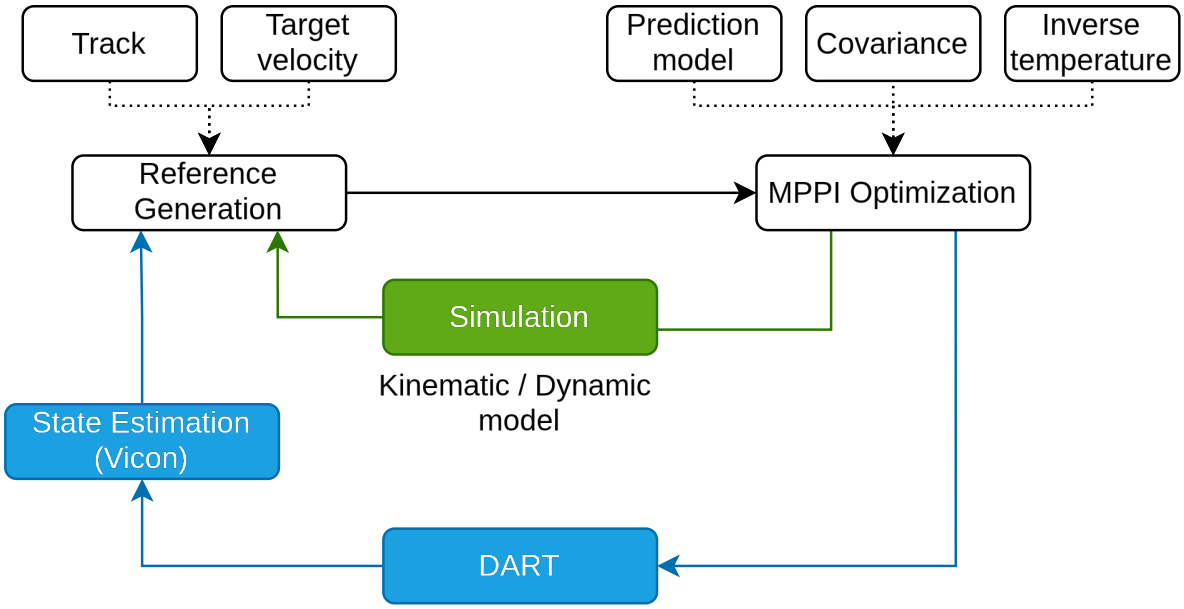



**Figure 3-1:** Real-time control architecture illustrating the interaction between state estimation, reference generation, MPPI optimization, and plant dynamics.

the dynamic model, respectively. Both models originate from prior system identification on the physical Delft's Autonomous-driving Robotic Testbed (DART) platform and are treated as fixed [33]. The underlying modeling assumptions were discussed in Chapter 2. The identification procedure is based on experimental data collected from the physical platform and employs gradient-based optimization to estimate parameters governing tire forces, steering dynamics, and actuation effects [33].

The control inputs consist of normalized throttle and steering commands defined on the interval $[-1, 1]$, where positive steering values correspond to left turns and negative values to right turns. These normalized commands are mapped to physical steering angles and longitudinal forces within the respective prediction models.

The employed prediction models are nonlinear and exhibit sensitivity to discretization when integrated over longer prediction horizons. Unlike many deterministic MPC implementations that rely on high-order numerical integrators or adaptive step-size schemes, the MPPI framework performs forward simulation using fixed-step integration. This choice is motivated by computational scalability. MPPI evaluates hundreds or thousands of rollouts per control update, each rollout must remain computationally inexpensive and amenable to parallel execution.

To maintain numerical stability without reducing the controller update interval, the prediction model is therefore integrated using multiple internal substeps within each control timestep $dt = 0.1$ s. The control input is held constant over this interval, while the state is propagated using a smaller internal integration step. Although this increases per-rollout cost, it preserves the controller update rate and real-time execution.

In simulation experiments, the same kinematic and dynamic models can also be used as plant models to generate closed-loop trajectories. This allows matched-model and mismatched-model studies by independently selecting the plant and prediction models. For these simula-

tions, the plant is integrated using a smaller timestep of 0.01 s to reduce numerical integration error.

## 3-3 MPPI Implementation Validation

After specifying the control architecture and prediction models in the previous section, the resulting MPPI implementation must be systematically validated on the DART platform. While Chapter 2 introduced MPPI at the algorithmic level, the present validation examines its behavior under the specific modeling and cost design choices introduced in this chapter. This section establishes a baseline evaluation of MPPI behavior on the considered platform rather than optimizing performance. Deviations from ideal reference tracking may arise from finite-sample effects or from structural interactions between the sampling distribution, prediction model, and cost function.

The focus of this section lies on the second category. Deviations from ideal reference tracking are expected and do not indicate implementation errors. Due to the sampling-based nature of MPPI, it must therefore be verified that any observed deviation from the reference is stochastic and unbiased, rather than the result of a systematic distortion in the control update mechanism. To isolate MPPI-intrinsic effects from modeling errors, the prediction model used within MPPI is also identical to the simulation model.

### 3-3-1 Model Asymmetry

When applying MPPI to a straight reference trajectory, a small but consistent lateral bias is observed, the vehicle settles slightly to the left of the desired path. This deviation, reproducible across runs, is independent of initial conditions, two of these runs are illustrated in Figure 3-2. The persistence of the effect indicates that it cannot be attributed to stochastic sampling noise, but instead reflects a systematic mechanism within the control pipeline. It therefore falls under the category of algorithm-intrinsic MPPI effects rather than finite-sample artifacts.

The origin of the lateral bias becomes apparent when examining the fitted steering model used in both simulation and MPPI rollouts. On the physical DART platform, the mapping from steering command to front wheel angle is nonlinear and asymmetric, such that steering commands of equal magnitude and opposite sign do not produce equal steering angles, and therefore do not yield symmetric lateral responses in the predicted trajectories. This mapping is also implemented in the already fitted vehicle models as a smooth combination of two hyperbolic tangent functions [33], with a weighting term $w_s(u)$ acting as a smooth switching function between parameterizations, visualized in Figure 3-3.

For a straight reference trajectory aligned with the longitudinal axis and a cost function that penalizes lateral deviation, this asymmetry induces a systematic imbalance in trajectory costs. In particular, if the steering mapping yields a larger lateral displacement for right-steering commands ($u < 0$) than for left-steering commands ($u > 0$) of equal magnitude, then for $u < 0$ one obtains $|y(-u)| > |y(u)|$ over a fixed rollout horizon. For a quadratic lateral tracking penalty $L_y = q_y y^2$, the corresponding running cost satisfies

$$L_y\left(y(-u)\right) > L_y\left(y(u)\right), \tag{3-7}$$

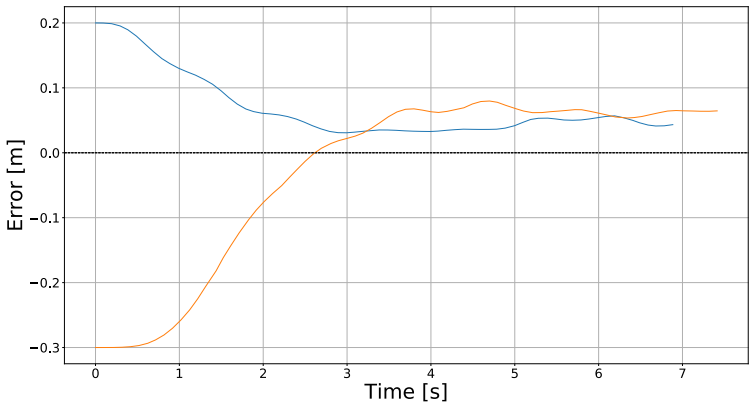



**Figure 3-2:** Two MPPI runs on a straight reference trajectory demonstrating a consistent lateral bias. The centerline denotes the reference, while the colored trajectories represent closed-loop vehicle paths.

and therefore MPPI assigns larger importance weights to the left-steering samples.

As a result, even if the sampling distribution in control space is symmetric about zero, the weighted distribution is not, and the MPPI mean update becomes biased, i.e. $u^*(0) \neq 0$. The sign of $u^*(0)$ is determined by the steering direction that yields systematically smaller lateral deviation, and therefore lower cost, under the asymmetric steering map.

This observation highlights that unbiased sampling in MPPI requires the prediction model to be symmetric with respect to signed steering commands. The experimentally identified steering model of the DART platform exhibits mild asymmetry due to actuator characteristics, which would otherwise introduce a bias between positive and negative steering perturbations. To avoid this effect within the MPPI rollouts, a symmetrized steering mapping is used in the prediction model. Specifically, the fitted nonlinear steering model, originally expressed as a weighted combination of two hyperbolic tangent functions, is evaluated with a constant blending weight, thereby removing sign-dependent asymmetry while preserving the overall nonlinear saturation behavior. This ensures that equal positive and negative steering commands yield symmetric steering angles during sampling.

The resulting desired steering angle $\delta^*$ is subsequently mapped back to a feasible steering command using an approximate inverse of the original nonlinear steering model

$$u = \begin{cases} \frac{1}{a_s} \operatorname{arctanh}\left(\frac{\delta^*}{b_s}\right) - c_s, & \delta^* \geq 0, \\ \frac{1}{e_s} \operatorname{arctanh}\left(\frac{\delta^*}{d_s}\right) - c_s, & \delta^* < 0, \end{cases} \tag{3-8}$$

followed by saturation to the admissible steering command interval. Where $a_s, b_s, c_s, d_s, e_s$ are constant parameters of the fitted model.

This approximation exploits the fact that the weighting function $w_s(u)$ transitions sharply between the two branches, allowing the inverse to be approximated by inverting the dominant hyperbolic tangent term based on the sign of the steering angle. With this inverse steering mapping the lateral bias vanishes. This provides a controlled reference MPPI performance

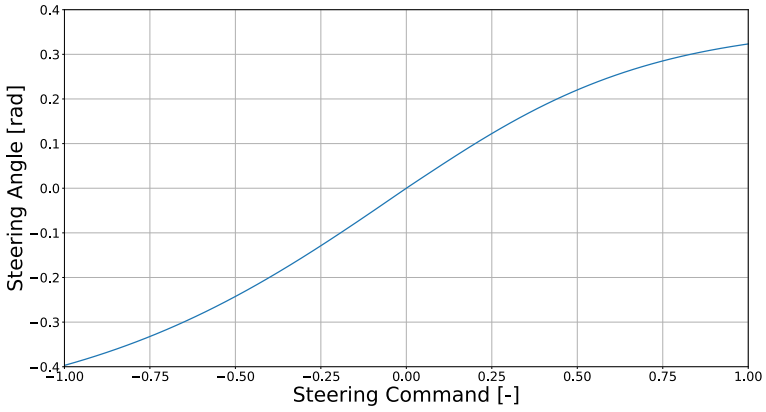



**Figure 3-3:** Nonlinear and asymmetric mapping from steering command to front-wheel steering angle identified for the DART platform.

against which subsequent controller modifications and experimental evaluations can be interpreted. Furthermore, the lateral bias presented here shows as a representative example of how asymmetric actuation or modeling can systematically distort MPPI updates.

## 3-4 Evaluation

The evaluation framework defines how controller performance is assessed across simulation and real-world experiments. In all experiments, the controller is tasked with following a predefined reference trajectory represented by a centerline embedded within a lane of fixed width. The centerline serves as the primary reference provided to the controller, while the lane boundaries define the admissible region for vehicle motion.

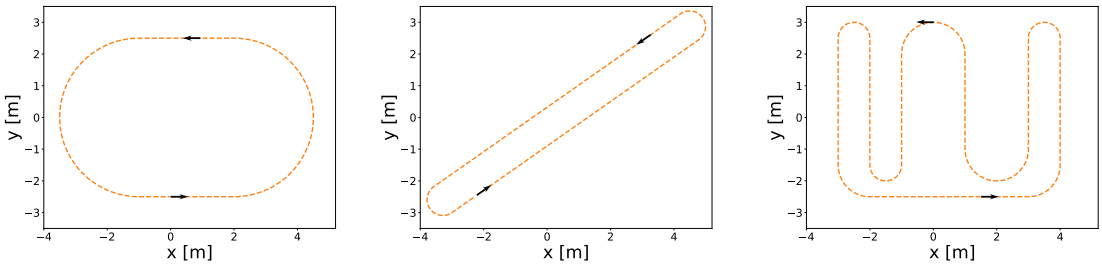
### 3-4-1 Scenarios

Controller performance is evaluated across a set of scenarios designed to progressively increase task complexity. These scenarios vary along three primary dimensions: reference geometry, reference speed, and plant fidelity. Each dimension targets specific aspects of closed-loop behavior, allowing controller properties to be examined in isolation before being combined in more demanding configurations.

Behavior is evaluated using a set of closed-loop track geometries, illustrated in Figure 3-4. The experimental scenarios are designed for an indoor testbed and therefore use track geometries that fit within the available laboratory footprint.

The first of these is a smooth Oval-track with long straights and gradual turns. This scenario is used to assess the controller's ability to negotiate sustained curvature while maintaining stability, as well as its ability to recover tracking performance on straight segments following cornering maneuvers. The second track referred to as the Sharp-Corner track, has a very long straight and a sharp turn on each end of the oval shape. This track isolates the behavior

induced by a sharp corner and has a long enough (longest possible in the laboratory) straight during which the oscillatory behavior and its convergence back to the reference can be studied. The final and most demanding scenario is the Racetrack, that combines multiple sharp corners with long straights. In this configuration, the vehicle is able to reach higher speeds before entering corners, significantly increasing the demands on predictive accuracy, actuation, and closed-loop stability. This scenario is therefore well suited for exposing limitations in controller behavior that may not be apparent in simpler environments. All these tracks have a lane width of 1 m.




**(a)** Oval-track **(b)** Sharp-Corner track **(c)** Racetrack

**Figure 3-4:** Track geometries used for controller evaluation, representing progressively increasing geometric complexity.

In addition to geometric complexity, reference speed is treated as a key parameter influencing task difficulty. Experiments are conducted at reference speeds ranging from 1.0 m/s to 3.5 m/s. Although these velocities are modest in absolute terms, the small track radii considered (down to $R = 0.5$ m) yield lateral accelerations on the order of $7.8 - 13.0$ m/s$^2$ at the higher reference speeds, since lateral acceleration scales according to $a_y = v^2/R$.

For comparison, Williams et al. [57] reported friction-limited behavior at speeds of $5-6$ m/s on a substantially larger test track with corner radii on the order of $10 - 15$ m. These conditions correspond to lateral accelerations of approximately $2.4-3.6$ m/s$^2$. Despite the lower absolute velocities used in the thesis, the markedly tighter curvature and reduced spatial margins result in comparable, or in terms of lateral loading, more demanding, dynamic conditions.

### 3-4-2 Obstacle Avoidance Scenarios

In addition to reference-tracking tasks with lane constraints, obstacle avoidance scenarios are included to evaluate the controller's ability to trade off path tracking accuracy and collision avoidance. An obstacle is placed along the nominal reference trajectory, requiring the controller to locally deviate from the reference while maintaining closed-loop stability and forward progress.

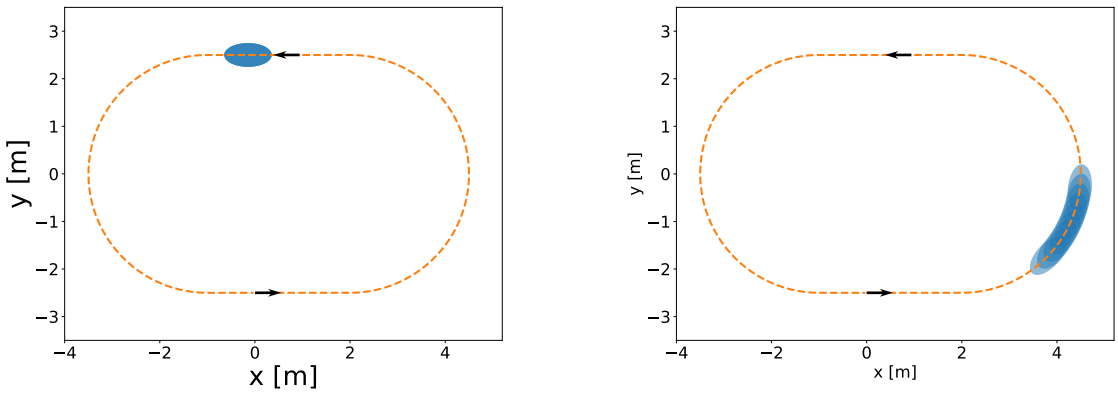
Two obstacle configurations are considered: a static obstacle that enforces a pronounced deviation from the nominal path, and a dynamic obstacle that moves along the reference trajectory at a constant speed and must be overtaken. In both cases, obstacle interaction is

modeled using an elliptical safety envelope in the $(x, y)$ plane, as explained in in Section 3-1-1. Ellipses are used as they allow anisotropic clearance by separately tuning longitudinal and lateral margins, reflecting that overtaking typically requires more longitudinal than lateral separation.

The envelope sizes are chosen as conservative interaction regions rather than physical obstacle footprints. Relative to the DART vehicle dimensions (wheelbase $l \approx 0.1735$ m and width $w \approx 0.08$ m), the envelopes are intentionally inflated to account for state estimation uncertainty, actuation variability, and tracking errors. This inflation additionally, ensures that obstacle avoidance requires a nontrivial maneuvering trade-off within the 1 m lane.

For the static obstacle, an envelope of ($a = 0.50$ m, $b = 0.25$ m) with fixed position and orientation is used, Figure 3-5a. With the obstacle centered on the lane, $b = 0.25$ m leaves $0.5 - 0.25 = 0.25$ m of lateral corridor to the lane boundary, forcing a decisive deviation while keeping the scenario feasible.

For the dynamic obstacle, a smaller envelope is used ($a = 0.40$ m, $b = 0.15$ m) to keep overtaking passable for all evaluated controllers and to avoid having obstacle avoidance dominate the evaluation, Figure 3-5b. Here, $b = 0.15$ m leaves $0.5 - 0.15 = 0.35$ m of lateral corridor, which reduces the likelihood of unavoidable constraint conflicts. The dynamic obstacle is modeled as a moving constraint with prescribed trajectory and speed rather than a full second-vehicle model with independently estimated yaw and dynamics. Accordingly, the envelope orientation is aligned with the local path tangent while the center $c(k)$ is propagated over the prediction horizon with a low constant speed.




**(a)** Static obstacle. **(b)** Dynamic obstacle

**Figure 3-5:** Elliptical safety envelopes used in obstacle avoidance scenarios, with time-varying envelope positions along the reference trajectory for the dynamic obstacle.

It is emphasized that this formulation does not aim to guarantee collision-free behavior. Since obstacle interaction is enforced solely through a soft cost evaluated at discrete rollout timesteps, violations may still occur under aggressive maneuvers, limited horizon length, or

unfavorable sampling realizations. These scenarios are therefore intended as controlled stress tests that expose how MPPI variants handle competing objectives and constraint proximity, rather than as a complete obstacle avoidance solution.

### 3-4-3 Metrics

To enable a systematic and objective comparison of controller behavior, performance is evaluated using a structured set of quantitative metrics. These metrics are chosen to jointly characterize reference tracking accuracy, dynamic stability, control smoothness, and overall driving performance, thereby reflecting both nominal behavior and transient responses under demanding conditions.

All metrics are computed at the controller update rate and evaluated over the full duration of each run. To assess driving performance in a racing context, the mean vehicle speed $\bar{v}$ is reported for all experiments. Reference tracking accuracy is evaluated using the Root Mean Square (RMS) lateral error with respect to the centerline. While the mean absolute lateral error reflects steady-state tracking performance, the RMS metric penalizes larger deviations more strongly and is therefore sensitive to intermittent excursions that typically occur during cornering or transient maneuvers.

Tracking consistency is evaluated using a Time-In-Bound (TIB) metric. Specifically, $\mathrm{TIB}_{<10\,\mathrm{cm}}$ is defined as the fraction of the run duration for which the absolute lateral error satisfies $|e_{\mathrm{lat}}(t)| < 10.0$ cm. This metric supports RMS error measures by capturing how consistently the vehicle remains close to the reference trajectory, rather than averaging over occasional large deviations. In cases where the vehicle exhibits larger tracking errors, a wider $\mathrm{TIB}_{<50\,\mathrm{cm}}$ bound is additionally reported. This threshold corresponds to the half-width of the lane and indicates whether the vehicle remains within the admissible driving region.

A metric that is used to quantify the number of samples that have at least a moderate influence in the final computed control input is the effective sample size

$$J_{\mathrm{eff}} = \frac{1}{\sum_j^J w_j^2}, \tag{3-9}$$

where $w_j$ are the normalized importance weights. This quantity measures the concentration of the weight distribution: if the weights are uniformly distributed ($w_j = 1/J$), then $J_{\mathrm{eff}} = J$, indicating that all samples contribute equally. Conversely, if the weight mass is concentrated on a small subset of trajectories, $J_{\mathrm{eff}}$ decreases toward 1. In the context of MPPI, $J_{\mathrm{eff}}$ therefore provides a direct indication of how many rollouts meaningfully influence the control update and serves as a practical diagnostic for tuning the temperature parameter $\lambda$ and the sampling covariance.

In some cases where the stability and control smoothness are of the essence, these are characterized using steering-related metrics. The RMS steering rate $\dot{\delta}_{\mathrm{RMS}}$ captures high-frequency variations in the control signal, while tire side-slip angle $\beta$ quantifies the extent to which lateral motion is induced by steering action.

To characterize transient stability following aggressive maneuvers, a side-slip settling-time metric is introduced in selected experiments. After a disturbance event (e.g., sharp corner exit or obstacle avoidance), the settling time is defined as the first time at which the moving

RMS envelope of the side-slip angle falls below 20% of its post-disturbance peak value and remains below this threshold for at least 0.5 s. This definition provides a robust time-domain measure of how rapidly lateral oscillations are attenuated.

For obstacle avoidance scenarios, tracking metrics are complemented with distance-based clearance measures taken over the entire run. While the minimum distance to the safety envelope $d_{\mathrm{E,min}}$, indicate if the safety region is penetrated, the minimum Euclidean distance to the obstacle center $d_{\mathrm{C,min}}$ provides a physically interpretable measure of avoidance relative to the actual vehicle footprint.

# Chapter 4

# Simulation Results

This chapter reports simulation results for the baseline Model Predictive Path Integral (MPPI) controller and several smoothness-oriented extensions. Unless stated otherwise, simulation experiments use the same cost function, reference generator, control limits, and control timestep ($dt = 0.1$ s), and evaluate a nominal configuration in which the controller prediction model matches the simulated plant. All metrics used in this chapter were defined in Section 3-4-3. Table values report averages over at least 15 laps. Time-series plots omit the initial convergence transient to emphasize steady-state closed-loop behavior.

The chapter proceeds in a structured manner. First, a parameter sensitivity analysis of the baseline MPPI controller is conducted to establish a well-performing reference configuration. Second, the baseline controller is compared against a Nonlinear MPC (NMPC) implementation to validate the correctness of the overall control architecture and model integration, consistent behavior across both controllers provides confidence in the implementation.

Subsequently, an oscillatory steering behavior observed in certain regimes is investigated through frequency-domain analysis to identify its dominant characteristics. After establishing a validated baseline, the smoothness-oriented MPPI extensions are individually verified and then compared systematically against baseline MPPI and each other across different tracks, reference speeds, and scenario configurations.

Finally, the performance implications of using the dynamic prediction model under reduced sampling budgets are examined, together with practical strategies to mitigate the resulting performance degradation.

## 4-1 MPPI Parameter Sensitivity Analysis

This section first examines baseline MPPI behavior on a simple reference-tracking task, and then quantifies how key MPPI parameters influence the exploration–exploitation trade-off and, in turn, the tracking performance and control smoothness. The prediction horizon, temperature, and sampling covariance are varied while keeping the reference trajectory and cost function fixed.

### 4-1-1 Speed Tracking Degradation

To isolate interactions between control channels, straight-line tracking with a constant reference speed $v_{\text{ref}}$ is considered. When only throttle is sampled, MPPI reduces to a one-dimensional control algorithm that tracks the desired speed. When throttle and steering are sampled jointly, however, the achieved speed $v$ is consistently biased below $v_{\text{ref}}$.

This systematic under-speeding is consistent with coupling introduced by importance weighting. Steering perturbations increase rollout cost variance, and high-throttle rollouts are more likely to amplify lateral deviations, leading to larger accumulated tracking error. Under importance weighting, these higher-cost rollouts receive smaller weights, which biases the weighted mean toward lower throttle even when lateral error remains small.

Figure 4-1 shows this bias by visualizing the speed tracking averaged over multiple runs and the maximum deviation around this average of single runs. Enabling steering sampling reduces the mean speed from 1.48 m/s to 1.42 m/s under identical reference conditions.

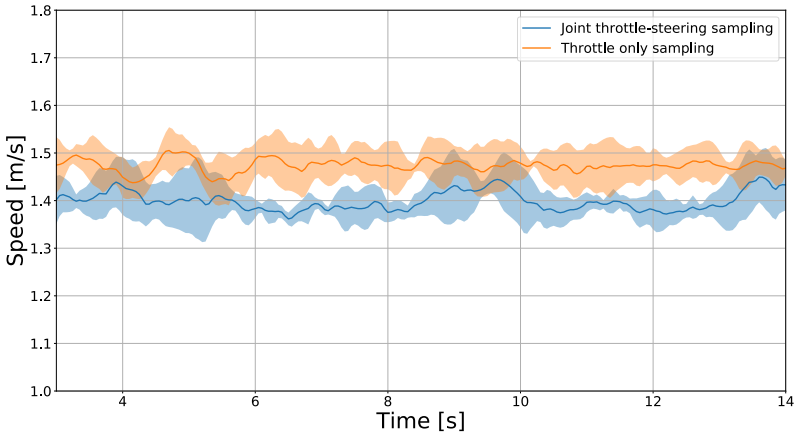
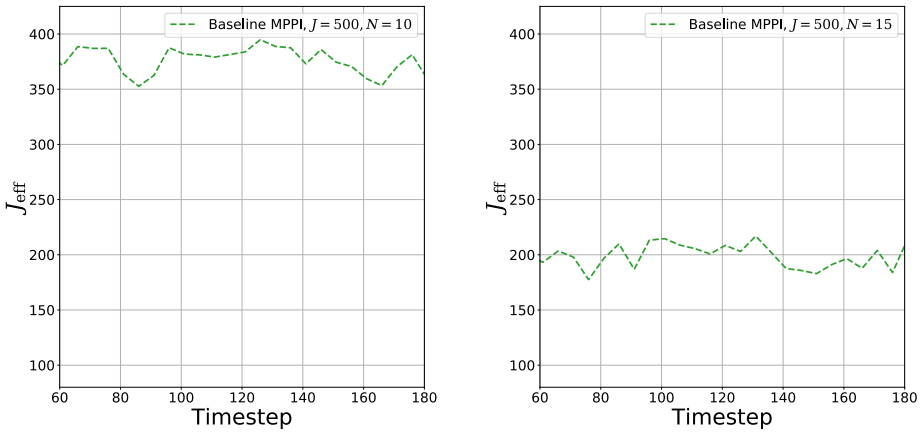



**Figure 4-1:** Speed tracking under throttle-only sampling and joint throttle–steering sampling for a straight-line task ($v_{\text{ref}} = 1.5, \text{m/s}$), illustrating systematic under-speeding under joint sampling.

### 4-1-2 Prediction Horizon Length

In the MPPI formulation, the prediction horizon $N$ sets the time window over which rollout costs are accumulated. Increasing $N$ enables longer-term planning but increases computational cost approximately linearly, thereby reducing the number of samples that can be evaluated within the fixed control period $dt = 0.1$ s. Moreover, longer horizons magnify cumulative cost differences between rollouts, causing trajectories that violate constraints or incur tracking error at any intermediate timestep to accumulate large total cost and receive small importance weights.This concentration manifests as weight degeneracy: only a small subset of trajectories contributes meaningfully to the control update. This concentration is quantified by the effective sample size $J_{\text{eff}}$, Eq. (3-9). In the Oval-track scenario (Figure 3-4a) with $v_{\text{ref}} = 1.5\,\text{m/s}$, $J = 500$, reducing the prediction horizon from $N = 15$ to $N = 10$




**(a)** Effective sample size $J_{\text{eff}}$ over time for $N = 10$.

**(b)** Effective sample size $J_{\text{eff}}$ over time for $N = 15$.

**Figure 4-2:** Effective sample size $J_{\text{eff}}$ over time for two prediction horizons on the Oval track ($J = 500$).

increases the mean $J_{\text{eff}}$ from approximately 197 to 376 (Figure 4-2), indicating substantially reduced degeneracy.

Figure 4-3 shows a representative run in which higher $J_{\text{eff}}$ coincides with a smoother speed response, consistent with reduced dominance of a small subset of rollouts. This illustrates a key distinction between sampling-based and deterministic predictive control: at fixed $J$, increasing $N$ can degrade MPPI performance because longer rollouts amplify weight degeneracy.

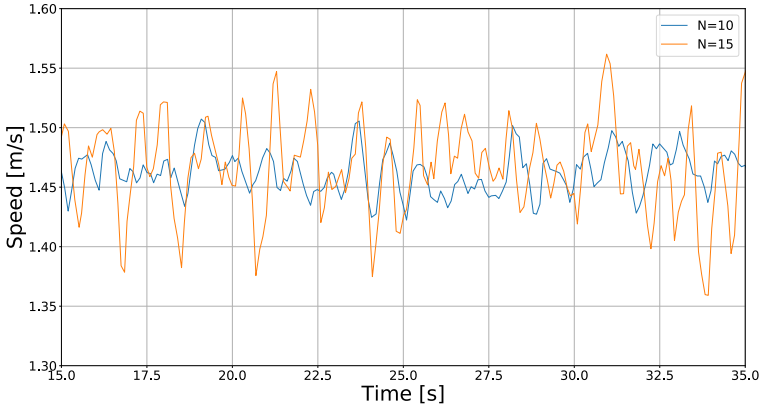



**Figure 4-3:** Speed response for prediction horizons $N = 10$ and $N = 15$ under identical conditions.

### 4-1-3 Temperature

The temperature parameter $\lambda$ governs how trajectory costs are mapped to importance weights and therefore controls the selectivity of the MPPI update. Lower values of $\lambda$ concentrate the weights on a small number of low-cost trajectories, causing a limited subset of control sequences to dominate the update. This increased selectivity enables the controller to approach, or even exceed, the reference speed despite exploratory steering perturbations. However, the resulting emphasis on a narrow set of samples can cause large variations in the selected control action between consecutive time steps, leading to spiky control inputs. In contrast, higher temperatures distribute the weights more uniformly across the sampled population.

This trade-off is evident in Figure 4-4, where the speed tracking is averaged over multiple runs and the maximum deviation from this average is shown. Although a lower temperature ($\lambda = 0.05$) relative to the temperature in the initial experiment ($\lambda = 0.5$) restores the mean speed to 1.48 m/s, it also induces pronounced oscillations in the speed, indicating reduced robustness and increased control effort.

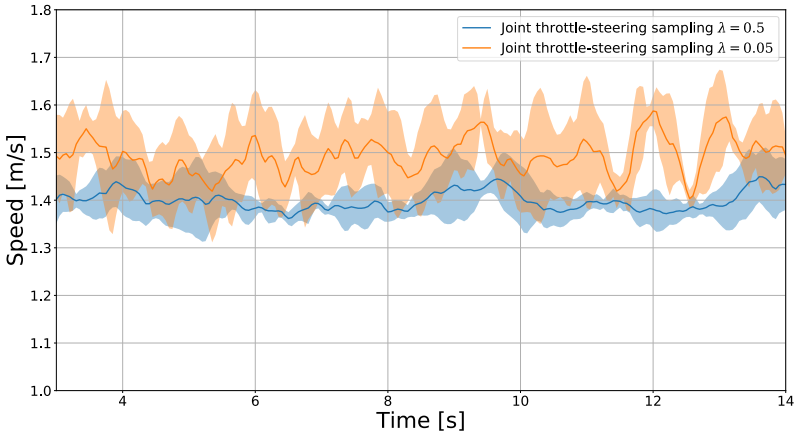



**Figure 4-4:** Speed tracking comparison for MPPI with steering sampling enabled at $\lambda = 0.05$ and $\lambda = 0.5$.

Table 4-1 summarizes the effects of the temperature parameter on tracking performance and control smoothness for both the Oval-track and the Racetrack (Figure 3-4c) configurations. Here, $\dot{\delta}_{\mathrm{RMS}}$ reflects steering smoothness, with lower values indicating reduced high-frequency control activity (Section 3-4-3).

**Table 4-1:** Influence of MPPI temperature on tracking performance and steering smoothness.

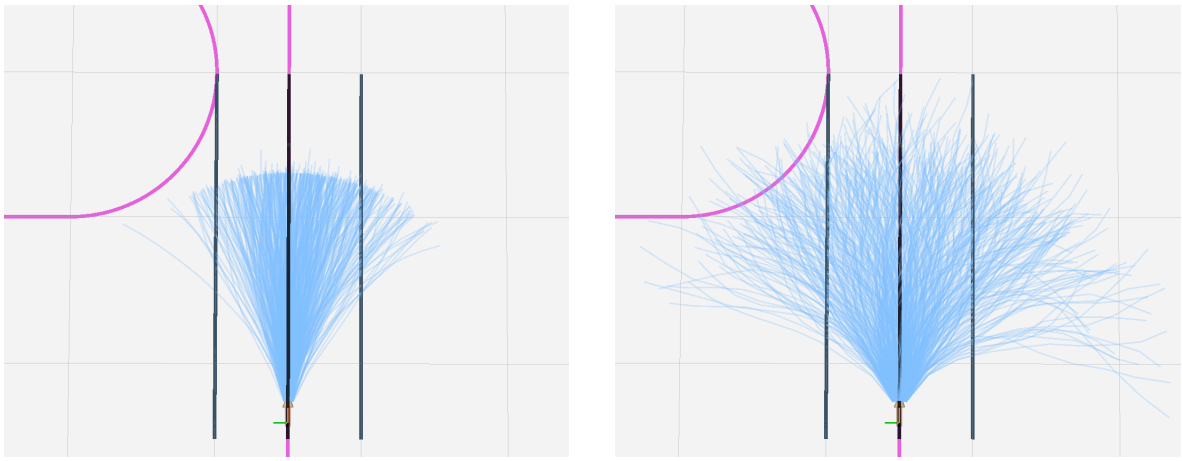
| Scenario | $\bar{v}$ $[\frac{\mathrm{m}}{\mathrm{s}}]$ | $e_{\mathrm{lat,RMS}}$ [m] | $\mathrm{TIB}_{<10\ \mathrm{cm}}$ [-] | $\dot{\delta}_{\mathrm{RMS}}[\frac{\mathrm{deg}}{\mathrm{s}}]$ |
|---|---|---|---|---|
| Oval-track, $v_{\mathrm{ref}} = 1.5, \lambda = 0.05$ | **1.51** | **0.019** | 1.000 | 13.94 |
| Oval-track, $v_{\mathrm{ref}} = 1.5, \lambda = 0.5$ | 1.40 | 0.037 | 0.998 | **4.31** |
| Racetrack, $v_{\mathrm{ref}} = 2.5, \lambda = 0.05$ | **2.55** | **0.029** | 0.999 | 38.18 |
| Racetrack, $v_{\mathrm{ref}} = 2.5, \lambda = 0.5$ | 2.37 | 0.037 | 0.995 | **27.70** |

Lower $\lambda$ improves tracking (higher mean speed and lower lateral error) but increases steering

rate activity, indicating less smooth control. Higher $\lambda$ reduces RMS steering rate substantially but produces a modest loss in tracking accuracy. The consistent trend across both tracks validates that $\lambda$ mediates the trade-off between tracking performance and control smoothness.

### 4-1-4 Sampling Covariance

In addition to the temperature parameter, the sampling covariance of the control perturbations plays a central role in shaping MPPI behavior. For the considered control architecture, this effect is governed by the throttle and steering covariances $\sigma_t$ and $\sigma_s$ respectively. Larger sampling covariances promote exploration but tend to increase control variability and degrade tracking performance. Conversely, a too small sampling covariance, which reduces control variability, can trap the sampling in locally suboptimal actions by limiting exploration. This effect is visualized in Figure 4-5, where $J = 500$ rollouts are shown as blue lines.



**(a)** Rollouts for tuned covariance ($\sigma_t = 0.1,\ \sigma_s = 0.2$).

**(b)** Rollouts for increased covariance ($\sigma_t = 0.3,\ \sigma_s = 0.5$).

**Figure 4-5:** Sampled rollout trajectories ($J = 500$) under two covariance settings, illustrating exploration spread.

Table 4-2 quantifies this trade-off on a high-speed Oval-track, comparing a tuned covariance setting against lower and higher covariances. The higher covariance shows an increase in mean speed, and a substantial increase in lateral error and an order-of-magnitude increase in the RMS steering rate. Lower covariance reduces exploration and smooths control, but may degrade tracking relative to the tuned configuration.

**Table 4-2:** Influence of sampling covariance on MPPI performance (Oval track, $v_{\text{ref}} = 2.5, \text{m/s}$).

| Covariance setting | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| Tuned ($\sigma_t = 0.1,\ \sigma_s = 0.2$) | **2.53** | **0.023** | **1.000** | **13.52** |
| Higher ($\sigma_t = 0.3,\ \sigma_s = 0.5$) | 2.56 | 0.068 | 0.944 | 94.23 |
| Lower ($\sigma_t = 0.05,\ \sigma_s = 0.1$) | 2.47 | 0.027 | 0.993 | 16.54 |

The effect of reducing the sampling covariance is further examined on the Racetrack with a high reference speed, with the results summarized in Table 4-3. Lower covariance values

lead to smoother steering behavior, as reflected by a reduced RMS steering rate, but are accompanied by increased lateral tracking error. This observation highlights a limitation when operating near the performance boundary: while reduced sampling covariance improves control smoothness, it may limit exploration of the aggressive steering–throttle combinations required for accurate tracking through sharp corners.

**Table 4-3:** Influence of sampling covariance on MPPI performance (Racetrack, $v_{\text{ref}} = 2.5, \text{m/s}$).

| Covariance setting | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| Tuned ($\sigma_t = 0.1,\ \sigma_s = 0.2$) | 2.55 | **0.029** | **0.999** | 38.18 |
| Lower ($\sigma_t = 0.05,\ \sigma_s = 0.1$) | **2.51** | 0.046 | 0.968 | **34.36** |

Taken together, the above discussions motivate the baseline MPPI configuration used in the remainder of this thesis. Unless stated otherwise, subsequent experiments use a prediction horizon length of $N = 10$, temperature $\lambda = 0.05$, and sampling covariances $(\sigma_t, \sigma_s) = (0.1, 0.2)$ with $J = 4000$ samples. These values are chosen to prioritize accurate reference tracking while maintaining sufficient exploration in high-curvature segments.

## 4-2 Comparison with Model Predictive Control

To contextualize MPPI performance, it is compared to a deterministic Model Predictive Control (MPC) baseline under identical reference trajectories, cost formulations, and control limits. First, nominal reference tracking is evaluated. Next, robustness is assessed through three real-world challenges: state-estimation noise, model mismatch, and steering delay. The NMPC controller used in this comparison is an existing DART controller implementation developed in prior work and is mentioned as MPC in the rest of the thesis for simplicity [33].

### 4-2-1 Baseline Performance Across Tracks

Table 4-4 summarizes results on two tracks (Oval-track and Racetrack) at two reference speeds. On the Oval-track, at 1.5 m/s MPC reduces the RMS lateral error, while also increasing mean speed. At 2.5 m/s, MPC again attains lower lateral error while both controllers maintain comparable mean speed. Across both tracks, MPPI exhibits higher steering rate activity than MPC. This is most notably at the Oval-track, where MPPI exhibits approximately twice the RMS steering rate, indicating much less smooth control and high frequency steering changes. On the Racetrack, MPC similarly exhibits lower lateral error at both speeds. At 2.5 m/s the MPPI runs on the Racetrack exhibit a substantially larger RMS lateral error while still remaining almost entirely within bounds ($\text{TIB}_{<10\text{ cm}} \approx 1$), indicating that the RMS is driven by a small persistent offset.

MPC consistently achieves lower lateral tracking error and maintains a higher mean speed than MPPI. This outcome is expected since MPC directly minimizes the defined objective over the prediction horizon. When the model is sufficiently accurate and the optimization problem remains feasible, MPC therefore produces high-performance trajectories with limited control jitter.

**Table 4-4:** Baseline performance comparison between MPC and MPPI under nominal conditions across tracks and reference speeds.

| Scenario | Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|---|
| Oval-track, $v_{\text{ref}} = 1.5$ | MPPI | 1.46 | 0.024 | 1.000 | 13.94 |
| Oval-track, $v_{\text{ref}} = 1.5$ | MPC | **1.51** | **0.015** | 1.00 | **6.51** |
| Oval-track, $v_{\text{ref}} = 2.5$ | MPPI | 2.53 | 0.023 | 1.000 | 16.55 |
| Oval-track, $v_{\text{ref}} = 2.5$ | MPC | 2.52 | **0.016** | 1.00 | **8.43** |
| Racetrack, $v_{\text{ref}} = 1.5$ | MPPI | 1.52 | 0.016 | 1.000 | 26.55 |
| Racetrack, $v_{\text{ref}} = 1.5$ | MPC | 1.53 | 0.011 | 1.000 | 22.79 |
| Racetrack, $v_{\text{ref}} = 2.5$ | MPPI | 2.53 | 0.111 | **0.996** | 38.18 |
| Racetrack, $v_{\text{ref}} = 2.5$ | MPC | 2.54 | **0.015** | 0.999 | 33.79 |

## 4-2-2 Robustness

The robustness studies in this section evaluate how MPC and MPPI degrade under three controlled imperfections that commonly affect real-world deployment: state-estimation uncertainty, model mismatch between prediction and plant, and steering-channel phase lag. Each test is implemented as a targeted stress perturbation applied while keeping the reference trajectory, and cost function fixed.

### State-Estimation Noise

To evaluate robustness to state-estimation noise, zero-mean noise was injected into the estimated vehicle state prior to each control update. The noise magnitudes were selected as a controlled stress test: large enough to produce measurable degradation, while avoiding trivial failure such as immediate loss of tracking or divergence. Specifically, the perturbations applied to the estimated state were

$$\begin{aligned} \sigma_{\text{pos}} &= 0.10 \text{ m}, \\ \sigma_{\psi} &= 0.05 \text{ deg}, \\ \sigma_{v} &= 0.05 \text{ m/s}, \quad \text{(applied to } v_x, v_y\text{)} \\ \sigma_{\omega} &= 0.02 \text{ rad/s}. \end{aligned}$$

Table 4-5 summarizes the resulting performance. On the Oval-track at $v_{\text{ref}} = 1.5\,\text{m/s}$, MPC retains higher lateral accuracy and remains fully within the $\pm 10\,\text{cm}$ bound. For both controllers, mean speed remains close to the nominal case, although steering activity increases substantially. On the Racetrack at $v_{\text{ref}} = 2.5\,\text{m/s}$, the degradation becomes more pronounced: lateral error increases and $\text{TIB}_{<10\text{ cm}}$ decreases for both controllers. Nonetheless, across both tracks the metrics remain relatively close to those obtained under ideal conditions without injected estimation noise. Overall, the injected noise affects MPPI more strongly in lateral tracking than MPC. Because the laboratory experiments rely on high-quality motion-capture feedback with comparatively small measurement noise, this analysis serves primarily as a stress test rather than a design constraint.

**Table 4-5:** Performance under injected state-estimation noise.

| Scenario | Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|---|
| Oval-track, $v_{\text{ref}} = 1.5$ | MPPI | 1.53 | 0.038 | 0.997 | 39.03 |
| Oval-track, $v_{\text{ref}} = 1.5$ | MPC | **1.51** | **0.017** | **1.000** | 36.32 |
| Racetrack, $v_{\text{ref}} = 2.5$ | MPPI | 2.58 | 0.149 | 0.809 | 89.74 |
| Racetrack, $v_{\text{ref}} = 2.5$ | MPC | 2.54 | **0.026** | **0.997** | 54.62 |

### Model Mismatch

In this experiment, the prediction model (kinematic model) used by the controller differs from the plant model (dynamic model) used in simulation, thereby introducing systematic prediction error. Table 4-6 summarizes performance at two reference speeds. At 1.5 m/s, both controllers achieve comparable mean speed and remain tightly to the reference.

**Table 4-6:** Performance on the Racetrack under model mismatch.

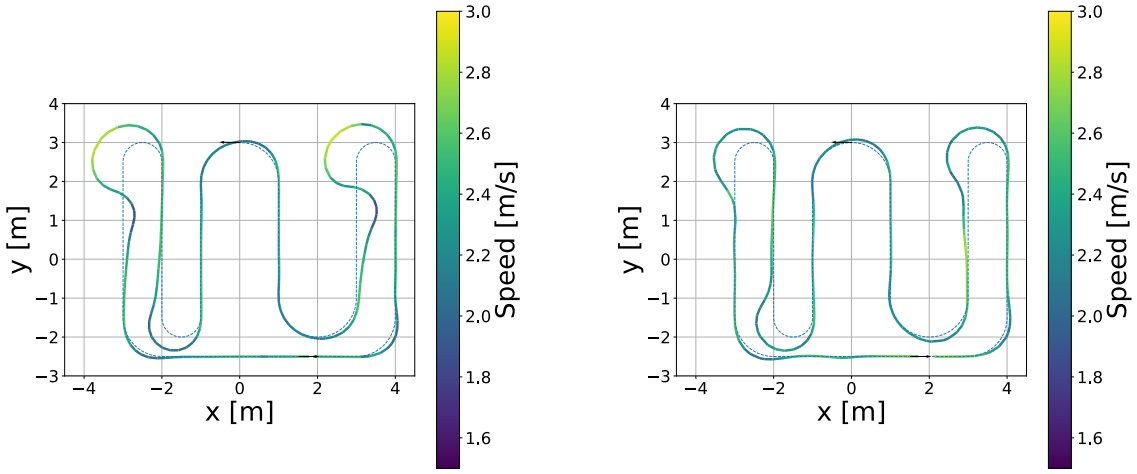
| Scenario | Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|---|
| Racetrack, $v_{\text{ref}} = 1.5$ | MPPI | 1.47 | 0.034 | 1.000 | 23.21 |
| Racetrack, $v_{\text{ref}} = 1.5$ | MPC | 1.44 | 0.025 | 1.000 | 24.91 |
| Racetrack, $v_{\text{ref}} = 2.5$ | MPPI | **2.40** | **0.086** | **0.799** | **44.25** |
| Racetrack, $v_{\text{ref}} = 2.5$ | MPC | 2.38 | 0.129 | 0.741 | 50.05 |

At $v_{\text{ref}} = 2.5$ m/s, model mismatch produces a clear degradation for both controllers: $\text{TIB}_{<10\text{ cm}}$ decreases and lateral error increases, indicating that tight tracking is no longer consistently achievable. Both controllers exhibit overshoot under model mismatch because the vehicle turns more slowly than predicted. As a result, trajectories approach the lane boundaries and require aggressive corrective steering to re-align with the reference. In this setting, MPPI attains lower lateral error and a modestly higher $\text{TIB}_{<10\text{ cm}}$ than MPC, suggesting improved tolerance to prediction error in terms of lane-keeping and trajectory continuity. MPC exhibits a larger increase in steering rate activity, indicating increased corrective effort as it attempts to maintain feasibility under a mismatched model. These trends are consistent with the a couple of paths visualized in Figure 4-6, where MPPI exhibits smoother recovery after sharp corners while maintaining comparable mean speed.

### Steering Delay

Experimental characterization of the DART steering actuator reports an effective steering delay of approximately 0.15 s, obtained via cross-correlation between a low-frequency sinusoidal steering command and the measured steering angle [33]. This identification captures the dominant phase lag of the steering system but does not distinguish between pure transport delay and dynamic effects such as actuator inertia, internal filtering, and rate limits. The reported 0.15 s value should therefore be interpreted as an effective delay summarizing aggregate steering-channel phase lag rather than as a strict dead time.

In simulation, this effect was conservatively modeled as a fixed steering command delay of 0.1 s in the plant, while both controllers operated at $dt = 0.1$ s. This abstraction neglects




**(a)** MPC **(b)** MPPI

**Figure 4-6:** Representative driven paths and speed profiles for MPC and MPPI under model mismatch on the Racetrack.

the fact that the real actuator typically responds immediately, albeit with limited bandwidth, and thus represents a worst-case approximation.

Results on the Oval-track at $v_{\text{ref}} = 1.5$ m/s are summarized in Table 4-7, with representative trajectories shown in Figure 4-7. Steering delay degrades tracking performance for both controllers. The degradation is substantially larger for MPC, which exhibits pronounced oscillatory behavior and reduced TIB. Furthermore, both controllers show a deterioration in tracking reliability, and the oscillatory trajectories in Figure 4-7 reflect a loss of stable reference tracking for MPC. These results suggest that delayed steering actuation imposes a fundamental limitation on closed-loop performance, rather than reflecting a controller-specific deficiency.

**Table 4-7:** Performance on the Oval track at $v_{\text{ref}} = 1.5, \text{m/s}$ under an injected $0.1, \text{s}$ steering delay.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| MPPI | 1.56 | **0.027** | **0.991** | 18.71 |
| MPC | 1.48 | 0.089 | 0.731 | 107.40 |

Under more demanding conditions with sharp corners and a high velocity, steering-channel phase lag induces pronounced oscillatory behavior in both controllers. For MPC, the combination of delayed steering and hard lane constraints can lead to loss of feasibility after a boundary violation, after which no valid control sequence is returned and the run must be terminated. MPPI also exhibits increased oscillations and occasional lane excursions under delay injection, however, because constraint handling is implemented through soft penalties, the controller continues to produce bounded control inputs and typically completes the

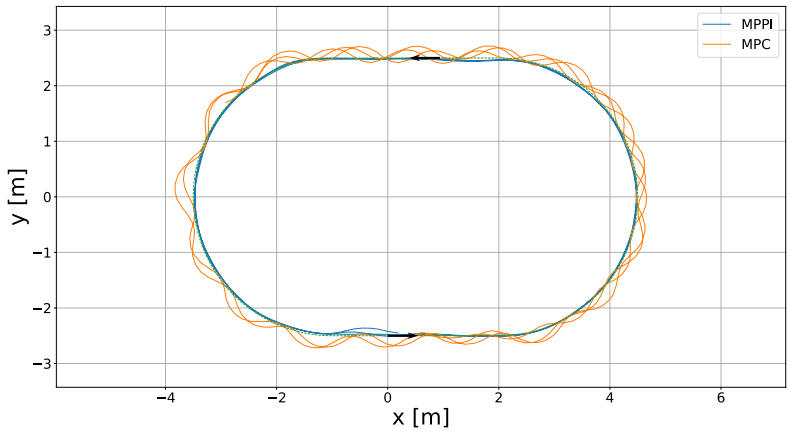



**Figure 4-7:** Driven paths for MPC and MPPI on the Oval track at $v_{\text{ref}} = 1.5, \text{m/s}$ with an injected $0.1, \text{s}$ steering delay.

lap. The paths driven by MPPI and MPC of a representative run are shown in Figure 4-8. This distinction highlights that steering-channel phase lag imposes a fundamental limitation on closed-loop tracking, while also exposing different failure modes associated with hard-constraint feasibility versus soft-constraint penalization.

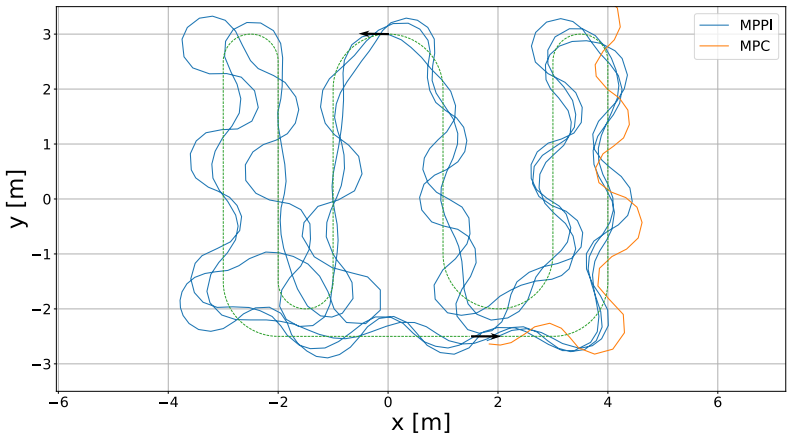



**Figure 4-8:** Driven paths for MPC and MPPI on the Racetrack at $v_{\text{ref}} = 2.5, \text{m/s}$ with an injected $0.1, \text{s}$ steering delay.

The robustness analysis indicates that MPPI exhibits greater tolerance to model mismatch and steering delay compared to the nonlinear MPC baseline. The improved robustness under model mismatch can be attributed to the sampling-based nature of MPPI. Because MPPI evaluates a distribution of perturbed control sequences at each update, it inherently considers a broader range of input variations around the nominal trajectory. As a result, control sequences that compensate for modeling inaccuracies or unmodelled dynamics may still receive significant weight in the importance sampling update.

## 4-3 Oscillatory Frequency Analysis

The delay-injection study in the previous section demonstrated that steering-channel phase lag induces sustained oscillatory behavior after sharp corners. While these oscillations are clearly visible in the time domain, it remains to be determined whether they represent a repeatable closed-loop dynamic mode or merely stochastic variability inherent to sampling-based control.

To distinguish between these possibilities, the steering command on the Sharp-Corner track is analyzed in the frequency domain, using the Fast Fourier Transform (FFT). The FFT is applied to the steering command time series recorded over a complete run. This transformation decomposes the time-domain control signal into its constituent frequency components, yielding a power spectrum that quantifies the relative contribution of different frequencies [39]. The logged steering command is cleaned by removing invalid samples, converted to steering angle, and demeaned to remove any constant offset. Using the average sample time inferred from the timestamps, a one-sided FFT is computed to obtain the magnitude spectrum. For each run, the steering command is segmented into fixed-length windows before and after the sharp corner. Identical window lengths are used across all runs to ensure consistent frequency resolution. The discrete Fourier transform of each window is computed, and the magnitude spectrum is analyzed. This procedure isolates the post-corner recovery phase while controlling for differences in trajectory length or transient alignment.

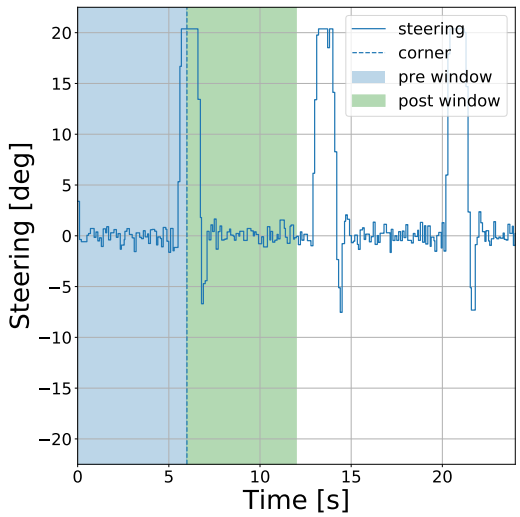
Figure 4-9 shows representative spectra for simulation without delay and with a $0.1\,\mathrm{s}$ steering delay. In the nominal case (no delay), the spectrum exhibits broadband low-amplitude content without a dominant frequency component. In contrast, when steering delay is introduced, a pronounced spectral peak emerges in the post-corner window.

This peak appears consistently across repeated runs, localized in a narrow frequency band, and is substantially weaker before the corner. This indicates that the oscillatory recovery is not a stochastic artifact of sampling but reflects excitation of a repeatable closed-loop mode induced by steering-channel phase lag. The emergence of a dominant spectral component under delayed actuation is consistent with classical feedback-system behavior: phase lag reduces effective damping and shifts the closed-loop poles toward the imaginary axis, resulting in underdamped oscillatory response [7]. The fact that this oscillation appears only after aggressive steering transients (e.g., sharp corners) further supports the interpretation that the system is temporarily driven near its stability boundary.

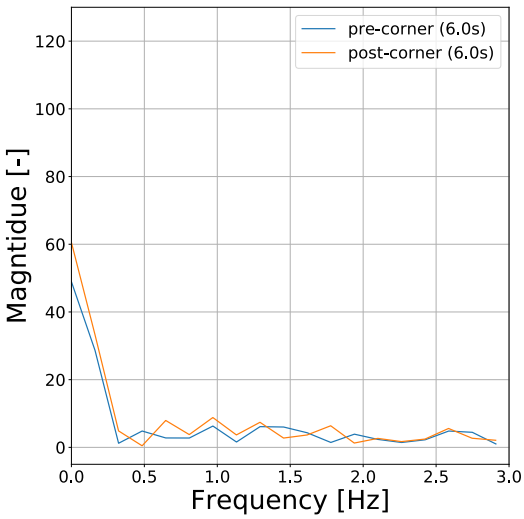
Thus, the frequency-domain signature provides a compact quantitative characterization of the oscillation mechanism. Because this spectral peak emerges solely when steering delay is present in simulation, it constitutes a falsifiable prediction: if real-world oscillations are primarily caused by steering-channel lag, a comparable dominant frequency should appear in experimental data. This prediction is tested in Chapter 5.

## 4-4 MPPI Extensions

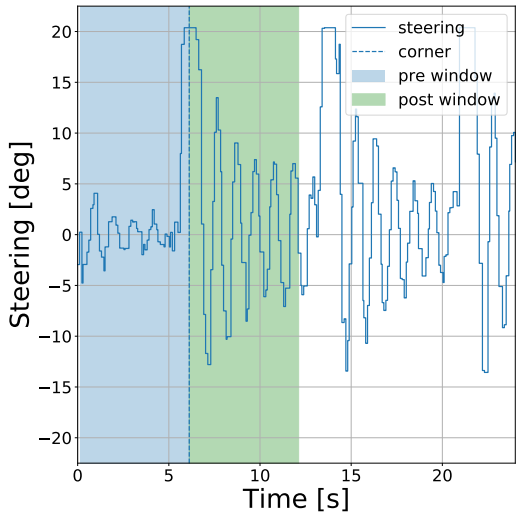
The jerky control behavior observed for baseline MPPI is not unique to the present implementation. Due to its sampling-based formulation, MPPI can generate control updates with substantial high-frequency content, especially under strongly selective weighting. As already



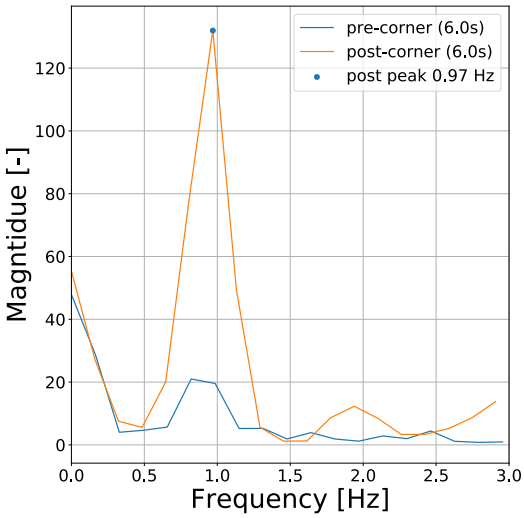

**(a)** Steering command in the time domain, with the pre- and post-corner analysis windows indicated (**no delay**).




**(b)** Steering command spectra computed over fixed windows before and after the sharp corner (**no delay**).




**(c)** Steering command in the time domain, with the pre- and post-corner analysis windows indicated ($0.1$ s **delay**).




**(d)** Steering command spectra computed over fixed windows before and after the sharp corner ($0.1$ s **delay**).

**Figure 4-9:** Frequency-domain characterization of post-corner steering behavior on the SharpCorner track for simulation without steering delay (top) and with a $0.1,\mathrm{s}$ delay (bottom).

seen when increasing $\lambda$ can reduce oscillations by flattening the importance weights, it typically also reduces achievable performance. This motivates extensions that explicitly target control smoothness while preserving the ability to react to fast changes.

This section evaluates three smoothness-oriented extensions: dynamic covariance MPPI,

Smooth MPPI (SMPPI), and Low-pass Filtered Sampling MPPI (LFS-MPPI). Each method modifies the sampling process or control parametrization. The extensions are first checked for mechanism-consistent behavior in simulation and then compared quantitatively. These configurations are then carried forward to the hardware evaluation in Chapter 5.

### 4-4-1 Extension Validation

The objective of this subsection is to verify that each MPPI extension exhibits the characteristic behavior it is theoretically designed to enforce. Validation is based on internal controller signals, including sampling covariance evolution, steering rate, effective sample size, and frequency content of the steering command. An extension is considered correctly implemented if its observed qualitative behavior is consistent with the underlying mechanism described in Section 2-4.

#### Dynamic Covariance MPPI

Dynamic covariance MPPI adapts the sampling covariance online using information from the importance weight distribution (Section 2-4-1), with the aim of reducing manual tuning sensitivity and adjusting exploration according to local task difficulty. In simulation, the dominant observed behavior is a progressive reduction of the steering covariance over the lap (i.e., $\sigma_s$ from 0.2 to approximately 0.01), with pronounced contractions immediately preceding high-curvature segments and partial recovery after corner exit. This behavior is most evident in the steering channel and is visualized in Figure 4-10, where the driven path is colored by the local steering covariance.

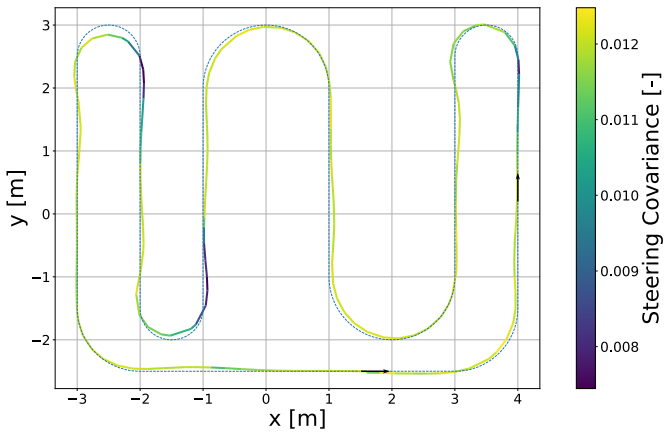



**Figure 4-10:** Spatial evolution of steering covariance $\sigma_s$ along the driven path in simulation with $v_{\text{ref}} = 2.5, \text{m/s}$.

This behavior confirms that the covariance adaptation mechanism responds to local task difficulty: in regions where only a small subset of trajectories yields low cost, the weighted covariance update contracts the sampling distribution, while exploration is relaxed elsewhere.

### Smooth MPPI

SMPPI modifies the control parametrization by sampling in a higher-order space (Section 2-4-2), typically control rates, and integrating these samples to obtain the control sequence. This effect is illustrated in Figure 4-11, which shows the sampled steering rate and the resulting integrated steering command.

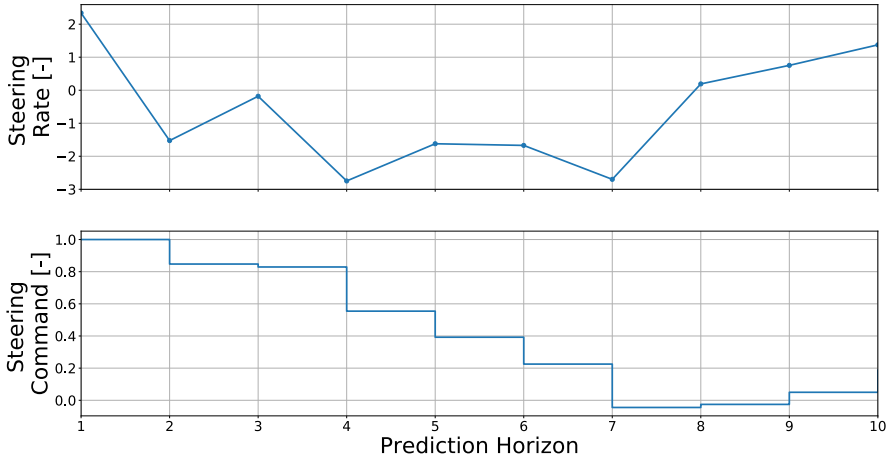
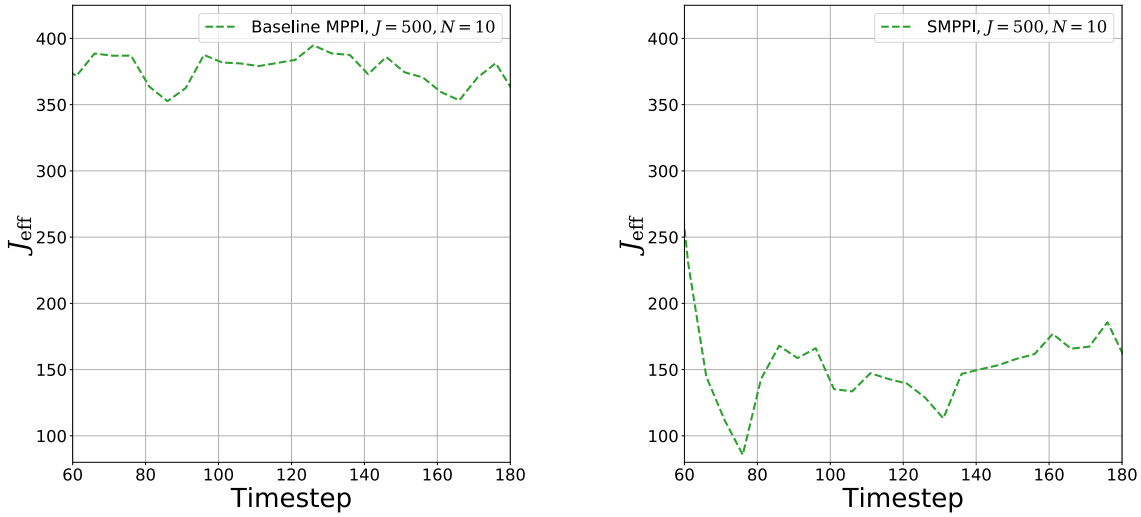



**Figure 4-11:** Sampled steering rate trajectories and corresponding integrated steering commands in SMPPI.




**(a)** Baseline MPPI. **(b)** SMPPI.

**Figure 4-12:** Effective sample size $J_{\text{eff}}$ over time for baseline MPPI and SMPPI on the Oval track ($J = 500$).

Sampling efficiency is however affected by SMPPI. One contributing factor is that SMPPI samples control rates without explicitly accounting for actuator saturation. Distinct rate

trajectories may therefore integrate to similar or identical saturated control sequences. This reduces the diversity of effectively distinct rollouts. The resulting reduction in effective sample size is illustrated in Figure 4-12, where SMPPI is compared against baseline MPPI under identical conditions.

### Low-pass Filtered Sampling MPPI

LFS-MPPI introduces temporal correlation directly into the sampled control perturbations (Section 2-4-3), shaping candidate trajectories prior to cost evaluation. Smoothness is therefore enforced at the source of the MPPI update.

Empirically, LFS-MPPI improves both control smoothness and sampling efficiency. By biasing the sampling process toward temporally coherent perturbations, a larger fraction of rollouts remains dynamically plausible.

Figure 4-13a compares the resulting control sequences and corresponding best trajectories for three cases: baseline MPPI, LFS-MPPI, and baseline MPPI with the final computed control sequence post-filtered using the same low-pass filter. All three results are generated from identical sampled perturbations, ensuring that differences arise solely from the treatment of the control signal rather than from stochastic variability. The comparison demonstrates that both filtering approaches reduce high-frequency fluctuations in the final control sequence, thereby attenuating jitter.

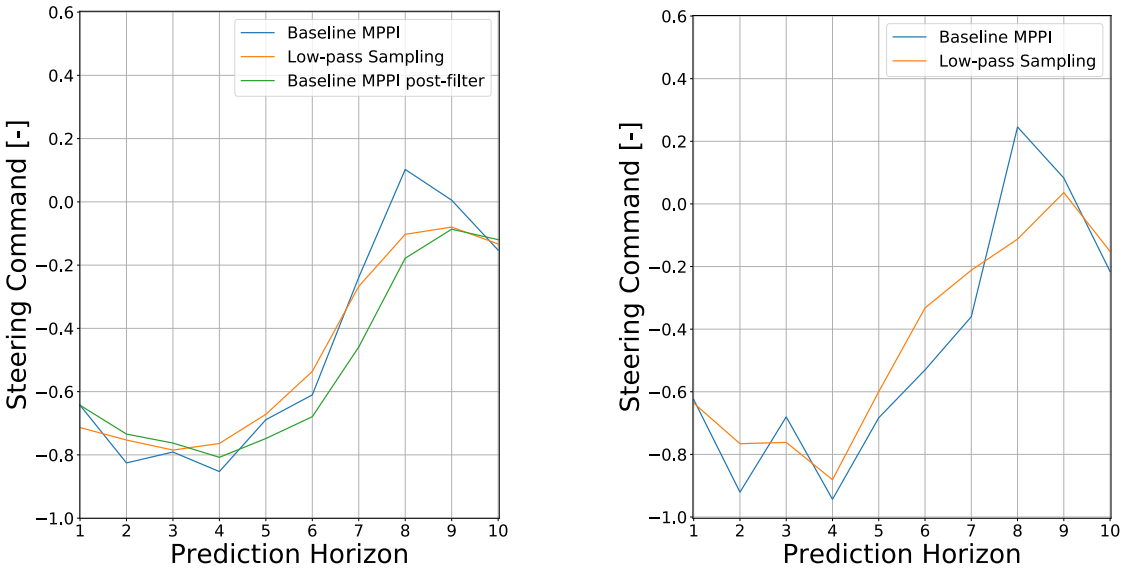
Figure 4-13b further illustrates the minimum-cost control sequences at the same time instant and for the same sample perturbations in both MPPI and LFS-MPPI. The lowest-cost sequence under LFS-MPPI is substantially less spiky, indicating that temporal correlation in the sampling process directly influences the structure of the optimal trajectory candidates.

An additional consequence of LFS-MPPI is increased robustness at reduced sample counts. Because temporally correlated perturbations yield a higher fraction of low-cost, dynamically feasible rollouts, comparable effective sample sizes can be achieved with fewer total samples. To assess this effect, baseline MPPI and LFS-MPPI were evaluated at similar values of $J_{\text{eff}}$.

At sufficiently large sample counts, both methods achieve comparable performance. However, when the total number of samples is reduced to levels at which baseline MPPI performance begins to degrade, LFS-MPPI maintains stable tracking behavior. This effect is quantified in Table 4-8. The results show that LFS-MPPI achieves comparable or improved performance relative to baseline MPPI while using half—or less—of the total number of samples. Notably, even when the effective sample size is lower than that of baseline MPPI, LFS-MPPI retains similar tracking performance and significantly reduced steering activity.

**Table 4-8:** Performance of MPPI and LFS-MPPI under reduced sample counts on the Oval track ($v_{\text{ref}} = 2.5, \text{m/s}$).

| Controller | $J_{\text{eff}}$ | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|---|
| Baseline MPPI ($J = 100$) | 2.680 | 2.593 | 0.046 | 0.965 | 45.60 |
| LFS-MPPI ($J = 50$) | 2.430 | 2.524 | 0.032 | 0.988 | 12.06 |
| LFS-MPPI ($J = 25$) | 1.828 | 2.504 | 0.036 | 0.990 | 12.64 |




**(a)** Computed control sequences under different filtering strategies.

**(b)** Trajectories with the lowest cost for baseline MPPI and LFS-MPPI.

**Figure 4-13:** Comparison of sampling and filtering strategies under identical perturbations: baseline MPPI, LFS-MPPI, and post-filtered control output.

### 4-4-2 Extension Comparison

With the individual extensions validated and their mechanisms and trade-offs established, this subsection presents a systematic comparison between baseline MPPI and three smoothness-oriented variants under matched simulation and experimental conditions. The objective is to assess how each method balances steering smoothness, tracking accuracy, and robustness as task difficulty increases.

#### Oval-track

The first set of experiments are performed on the Oval-track at $v_{\text{ref}} = 1.5$ m/s. Averaged results over multiple runs are reported in Table 4-9. All controllers track the reference accurately in this setting: all variants remain within the 10 cm lateral bound for the full run while maintaining mean speed close to $v_{\text{ref}}$. Thus, observed differences in steering activity are not explained by reduced speed. Relative to baseline MPPI, all extensions reduce the RMS steering rate, confirming that the smoothing mechanisms act as intended.

Among the extensions, LFS-MPPI yields a small but consistent improvement in lateral tracking accuracy, achieving the lowest RMS lateral error. This is consistent with the increased sampling efficiency discussed above. However, absolute differences in lateral error remain small in this low-speed, low-curvature scenario, and strong conclusions about relative tracking performance are therefore premature.

**Table 4-9:** Performance on the Oval-track in simulation at $v_{\text{ref}} = 1.5$ m/s.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| MPPI | 1.46 | 0.024 | 1.000 | 13.94 |
| Dyn-cov MPPI | 1.52 | 0.027 | 1.000 | 9.18 |
| SMPPI | 1.51 | 0.031 | 1.000 | **5.13** |
| LFS-MPPI | 1.51 | **0.018** | 1.000 | 6.07 |

### Racetrack

To further stress the controllers, task difficulty is increased by raising the reference speed to $v_{\text{ref}} = 2.5$ m/s and switching from the Oval-track to the Racetrack. Results are summarized in Table 4-10. For baseline MPPI, higher speed and curvature increase lateral error and substantially increase steering activity, reflected by the large RMS steering rate. In contrast, all extensions reduce RMS lateral error relative to MPPI while also reducing the RMS steering rate. Thus, in this scenario, smoothness-oriented modifications improve both control smoothness and tracking accuracy.

A more nuanced difference appears in the TIB metric. Dyn-cov MPPI and SMPPI achieve lower RMS lateral error than baseline MPPI but slightly lower TIB. This indicates that baseline MPPI remains within the 10 cm bound for most of the run but occasionally exhibits larger deviations (consistent with aggressive corrective actions), whereas Dyn-cov MPPI and SMPPI produce longer periods slightly outside the threshold. LFS-MPPI provides the most balanced performance. It achieves the lowest RMS lateral error while maintaining perfect TIB and a steering rate RMS comparable to the other extensions.

**Table 4-10:** Performance on the Racetrack in simulation at $v_{\text{ref}} = 2.5$ m/s.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| MPPI | 2.53 | 0.11 | 0.996 | 38.18 |
| Dyn-cov MPPI | 2.50 | 0.038 | 0.986 | 31.09 |
| SMPPI | 2.54 | 0.055 | 0.910 | 29.60 |
| LFS-MPPI | 2.52 | **0.020** | **1.000** | **29.11** |

### Obstacle Avoidance

A static obstacle is introduced by augmenting the running cost with a soft penalty, modeled as an ellipse with semi-axes $a = 0.50$ m and $b = 0.25$ m and a fixed orientation.

Table 4-11 shows that all controllers maintain strictly positive minimum clearance with respect to the safety envelope, indicating that the penalty is effective in this simulation scenario. Moreover, all MPPI extensions reduce steering activity relative to baseline MPPI while preserving comparable mean speed. Given an effective diagonal footprint of approximately 0.30 m for the DART vehicle (corresponding to a conservative physical radius of about 0.15 m), all controllers maintain ample separation from the actual obstacle.

A second experiment considers a moving obstacle, with $a = 0.40$ m, $b = 0.15$ m, where both the center position and orientation of the safety envelope vary over the prediction horizon

**Table 4-11:** Static obstacle avoidance in simulation on the Oval-track at $v_{\text{ref}} = 2.5$ m/s.

| Controller | $\bar{v}$ [m/s] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ | $d_{\text{E,min}}$ [m] | $d_{\text{C,min}}$ [m] |
|---|---|---|---|---|
| MPPI | 2.54 | 34.36 | 0.042 | 0.338 |
| Dyn-cov MPPI | 2.42 | 23.43 | 0.068 | **0.362** |
| SMPPI | 2.52 | **15.87** | 0.068 | 0.327 |
| LFS-MPPI | 2.52 | 23.82 | 0.059 | 0.348 |

along the reference trajectory with $v = 1.0$ m/s. The results in Table 4-12 show that, in this more challenging scenario, all controllers exhibit partial penetration of the safety envelope, reflected by negative minimum envelope clearance. Under the adopted interpretation, this indicates a temporary reduction in safety margin with respect to the inflated envelope rather than a physical collision. Such behavior is consistent with the use of a soft penalty, a finite prediction horizon, and the additional difficulty introduced by obstacle motion.

**Table 4-12:** Dynamic obstacle avoidance in simulation on the Oval-track at $v_{\text{ref}} = 2.5$ m/s. Negative envelope clearance indicates penetration of the inflated safety envelope. The minimum center distance $d_{\text{c,min}}$ provides a physically interpretable measure of obstacle separation.

| Controller | $\bar{v}$ [m/s] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ | $d_{\text{E,min}}$ [m] | $d_{\text{C,min}}$ [m] |
|---|---|---|---|---|
| MPPI | **2.49** | 26.50 | -0.060 | **0.235** |
| Dyn-cov MPPI | 2.46 | **10.40** | -0.087 | 0.221 |
| SMPPI | 2.43 | 14.84 | -0.092 | 0.216 |
| LFS-MPPI | 2.41 | 25.57 | -0.098 | 0.194 |

Crucially, when considering the minimum distance to the obstacle center, all controllers still maintain separation comparable to or larger than the effective physical radius of the DART vehicle. Even in the most challenging case, the vehicle does not approach the obstacle center closer than 0.194 m, supporting the conclusion that physical avoidance is preserved despite partial envelope violation.

### Model Mismatch

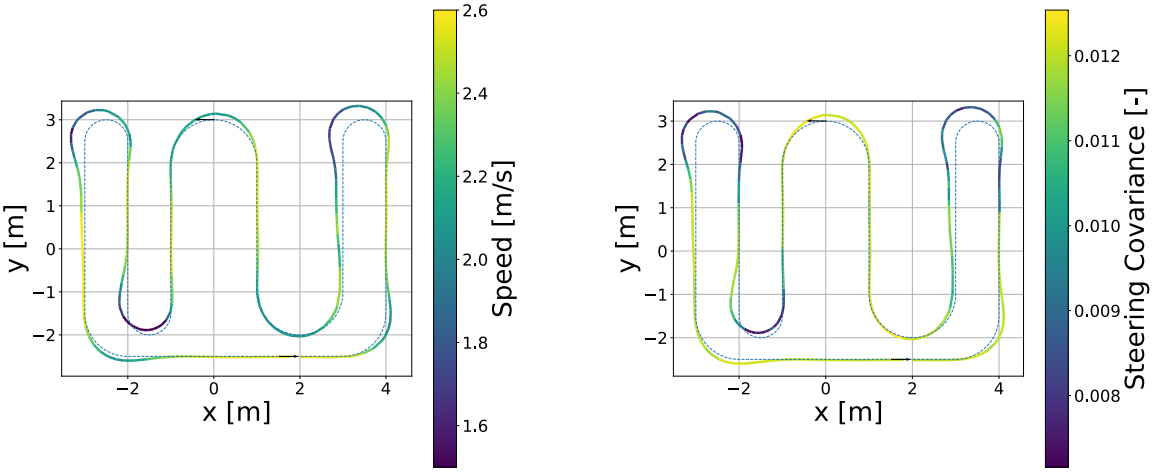
Robustness to model mismatch is assessed by evaluating the controllers with the kinematic prediction model while simulating the plant using the dynamic model. This evaluation is performed in the most challenging scenario, namely the Racetrack at $v_{\text{ref}} = 2.5$ m/s, with results summarized in Table 4-13.

**Table 4-13:** Performance on the Racetrack in simulation under model mismatch at $v_{\text{ref}} = 2.5$ m/s.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| MPPI | 2.40 | 0.086 | 0.799 | 44.25 |
| Dyn-cov MPPI | **1.80** | 0.090 | 0.690 | 29.31 |
| SMPPI | 2.32 | 0.073 | 0.844 | 30.47 |
| LFS-MPPI | 2.39 | 0.068 | **0.870** | 34.81 |

Overall, the extensions exhibit qualitatively similar degradation under model mismatch. The

most pronounced deviation occurs for Dynamic Covariance MPPI, which achieves a substantially lower mean speed than the other controllers. This reduction is not due to an inability to reach $v_{\text{ref}}$ in principle, but instead arises from its behavior in a subset of sharp corners. For Dynamic Covariance MPPI, this situation coincides with locally reduced steering covariance, as illustrated in Figure 4-10. Consequently, sufficiently large steering corrections are undersampled, limiting the controller's ability to recover at high speed. To avoid lane boundary violations, the controller instead reduces speed until recovery becomes feasible. This behavior is visualized in Figure 4-14, where reduced speed and low steering covariance occur in close proximity to the lane boundaries.




**(a)** Spatial evolution of speed.

**(b)** Spatial evolution of steering covariance.

**Figure 4-14:** Driven path for Dynamic Covariance MPPI under model mismatch, illustrating reduced speed and contracted steering covariance near lane boundaries.

### Steering Delay

To evaluate robustness against actuation latency, all controllers are tested on the Sharp-Corner track at $v_{\text{ref}} = 1.5$ m/s with an injected steering delay of 0.1 s. This configuration consistently induces oscillatory steering behavior in baseline MPPI and therefore serves as a representative stress scenario for assessing transient stability.

**Table 4-14:** Settling time on the Sharp-Corner track at $v_{\text{ref}} = 1.5$ m/s with a steering delay of 0.1 s.

| Controller | settling time $\beta$ [s] |
|---|---|
| MPPI | 4.662 |
| Dyn-cov MPPI | 2.094 |
| SMPPI | 2.310 |
| LFS-MPPI | 2.407 |

Table 4-14 reports the side-slip settling time following corner exit, while Figure 4-15 illustrates the corresponding time-domain response of the side-slip angle $\beta$. The settling time is defined as described in Section 3-4-3.

Baseline MPPI frequently fails to fully settle before the subsequent corner, and its reported value should therefore be interpreted as an upper bound within the considered observation window rather than as a true steady-state settling time. In contrast, all three smoothness-oriented extensions significantly reduce the settling time, indicating faster attenuation of lateral oscillations under delayed actuation.

The results suggest that limiting high-frequency steering activity improves robustness to steering delay. By reducing rapid steering reversals and oscillatory control inputs, the extensions mitigate the phase-lag amplification introduced by the delay, resulting in more stable lateral dynamics.

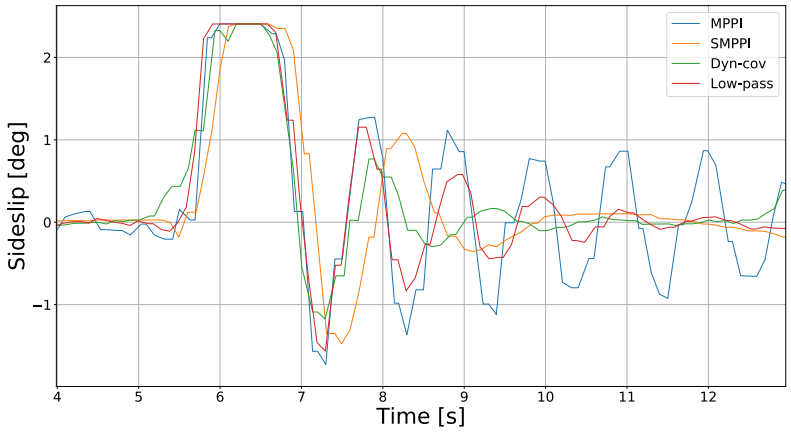



**Figure 4-15:** Side-slip angle response for each MPPI variant under a $0.1, \mathrm{s}$ steering delay, illustrating differences in oscillatory decay.

## 4-5 Dynamic Prediction Model

Up to this point, MPPI has employed the kinematic model for prediction, primarily because its low computational cost permits large sample counts within the real-time control period $dt = 0.1$ s. The dynamic model provides a more faithful representation of transient lateral dynamics and tire-force saturation near the handling limit, but at substantially higher computational cost. Under a fixed compute budget, this increased fidelity reduces the number of rollouts that can be evaluated per control update.

Replacing the kinematic prediction model with the dynamic model renders the baseline configuration with $J = 4000$ rollouts infeasible in real time. To satisfy the $dt = 0.1$ s deadline, the sample count must be reduced to approximately $J = 500$ when using the dynamic model. Since this is the number of samples under which the computational time is never saturated.

Consistent with the sampling-efficiency analysis in Section 4-1-2, reducing the number of rollouts $J$ increases control variance. This effect is illustrated for a nominal Oval-track scenario

at $v_{\text{ref}} = 1.5$ m/s in Table 4-15. While lateral tracking performance remains comparable, the RMS steering rate increases markedly as $J$ decreases.

**Table 4-15:** Effect of rollout count $J$ on steering smoothness, using the dynamic model as the prediction model, on the Oval-track, with $v_{\text{ref}} = 1.5$ m/s.

| $J$ | $\dot{\delta}_{\text{RMS}}$ [deg/s] |
|---|---|
| 4000 | 13.94 |
| 2000 | 17.87 |
| 1000 | 21.64 |
| 500 | 27.94 |

Although the dynamic prediction model improves physical fidelity, the accompanying reduction in sampling quality under a fixed computational budget increases control variability. Higher model fidelity alone therefore does not guarantee improved closed-loop performance.

Temporally correlated sampling alters this trade-off. By filtering the sampled perturbations prior to rollout, low-pass filtered sampling increases the fraction of dynamically plausible, low-cost trajectories at reduced $J$ (Table 4-8). As a result, the dynamic prediction model becomes real-time deployable at $J = 500$ while maintaining stable and smooth closed-loop behavior, achieving an RMS steering rate of 2.87 deg/s on the same track.

This observation highlights a central insight: in sampling-based predictive control under real-time constraints, model fidelity and sampling efficiency are intrinsically coupled. Increasing model fidelity is beneficial only if sufficient effective sampling quality is preserved. The laboratory experiments in Section 5-4 evaluate this coupling under physically realized constraints.

# Chapter 5

# Laboratory Results

This chapter evaluates Model Predictive Path Integral (MPPI) and the proposed smoothness-oriented extensions on the physical Delft's Autonomous-driving Robotic Testbed (DART) platform. The laboratory experiments in this chapter make use of the same evaluation protocol, cost structure, and reference trajectories as in Chapter 4, but expose the controllers to real sensing, actuation, and timing effects.

The objective of this chapter is to assess whether the mechanisms identified in simulation persist under real-world conditions. Particular emphasis is placed on tracking accuracy, steering oscillations, and recovery behavior after aggressive maneuvers. This is consistent with the evaluation philosophy introduced in Chapter 1, where the emphasis is placed on assessing the intrinsic robustness of MPPI-based control under laboratory imperfections.

This chapter first introduces the experimental testbed. Subsequently, the baseline MPPI performance in the laboratory is analyzed, including the identification and characterization of its dominant shortcomings. The proposed extensions are then evaluated, with specific attention to the reactivity-smoothness trade-off. The chapter concludes with an assessment of the maximum reliably achievable speed when employing the dynamic prediction model, leveraging the sample-efficiency benefits of temporally correlated sampling identified in simulation.

## 5-1 Experimental Setup

This section describes the hardware and state-estimation pipeline used for all laboratory experiments. As already mentioned the design mirrors the simulation pipeline explained in Section 3-2. This alignment ensures that differences between simulation and laboratory results arise primarily from real-world sensing and actuation effects, rather than software differences.

### 5-1-1 Delft's Autonomous-driving Robotic Testbed

The experimental evaluation is conducted using DART (Figure 5-1), a small-scale Autonomous Vehicle (AV) platform developed for control and planning research [33]. The platform is ex-

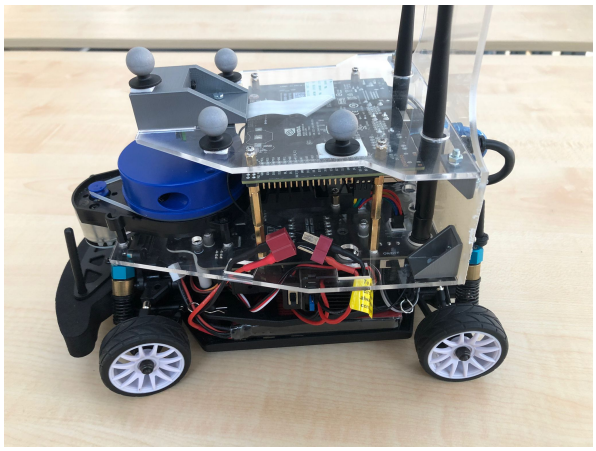


**Figure 5-1:** Delft's Autonomous-driving Robotic Testbed platform used for laboratory evaluation [33].

plicitly designed to be low-cost and reproducible, allowing experiments to be conducted in the laboratory without the infrastructure and safety constraints associated with full-scale vehicles.

Control commands are computed on an external workstation and transmitted wirelessly to the vehicle via ROS communication. The onboard JetRacer Pro AI hardware converts normalized control inputs into steering and propulsion commands. The measured command transmission latency is approximately 5 ms, which is small relative to the controller timestep used ($dt = 0.1$ s). Throttle actuation delay has been experimentally observed to be negligible. In contrast, steering actuation exhibits a measurable delay that cannot be neglected and will be explicitly analyzed in the subsequent sections [33].

### 5-1-2 Mobile Robotics Lab

Closed-loop control of the DART platform relies on external state measurements provided by the Mobile Robotics Lab at TU Delft. The laboratory is equipped with a Vicon motion capture system that uses multiple reflective markers mounted on the chassis. This system provides position and orientation measurements at an update rate of approximately 100 Hz. These position measurements are then used to estimate the vehicle body-frame velocities in the same way as in simulation, as described in Section 3-1-3.

## 5-2 Laboratory Oscillatory Behavior

When deploying MPPI on the DART platform, substantially stronger lateral oscillations are observed compared to nominal simulation. In the laboratory, oscillations are most pronounced in the steering command and coincide with spikes in side-slip angle, particularly during and after corner exit. This effect is already pronounced at $v_{\text{ref}} = 1.5$ m/s on the Oval-track, as is visualized in Figure 5-2.

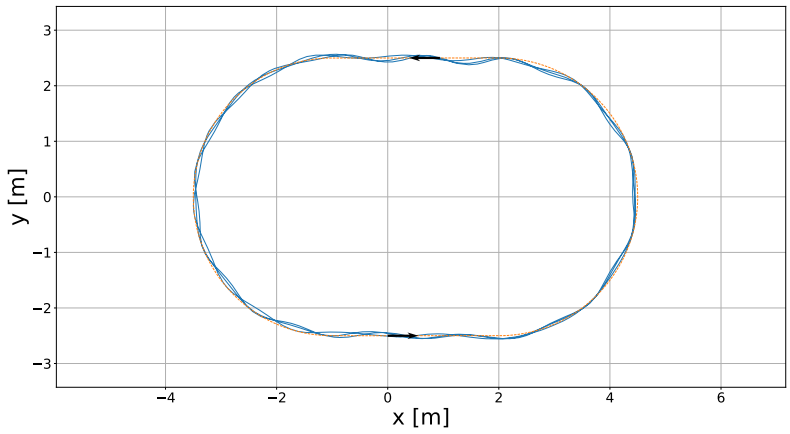



**Figure 5-2:** Representative laboratory trajectory under baseline MPPI, illustrating oscillatory lateral behavior during cornering.

### 5-2-1 Time-Domain Behavior

Across all laboratory experiments, increased steering activity is consistently associated with reduced closed-loop stability. This effect is most pronounced when driving sharp corners or operating at higher reference speeds. In these regimes, the DART exhibits oscillations in the steering command that are closely coupled with oscillations in yaw rate and side-slip angle. Compared to smoother track configurations, the Sharp-Corner layout, with its tighter curvature, amplifies these effects, resulting in prolonged oscillatory behavior after corner exit rather than rapid convergence to the reference trajectory. An illustrative example is shown in Figure 5-3.

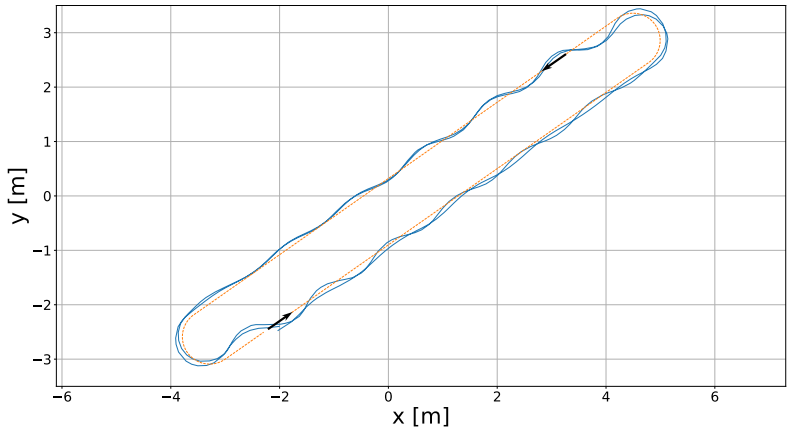



**Figure 5-3:** Laboratory trajectory on the Sharp-Corner track at $v_{\text{ref}} = 1.5, \text{m/s}$, illustrating prolonged post-corner oscillations.

To assess whether steering delay is the primary contributor to the observed oscillations, and

to determine whether smoother steering control can mitigate this behavior, several controller-level interventions were evaluated in laboratory experiments.

Firstly, as shown in Section 4-1-3, increasing the MPPI temperature reduces the RMS steering rate by flattening the importance-weight distribution. This reduces the dominance of a small number of aggressive rollouts and results in smoother control updates.

Secondly, Section 4-1-4 demonstrates that reducing the sampling covariance produces a similar stabilizing effect. By directly limiting the magnitude of sampled perturbations, step-to-step steering variability is constrained, thereby attenuating oscillatory behavior.

Thirdly, a Savitzky–Golay filter was applied to the computed control sequence (Section 2-3-3) to suppress high-frequency steering components. While this post-filtering approach effectively reduces oscillations, it was employed solely for diagnostic purposes. Because it modifies the control signal after the MPPI importance-weight update, it violates the algorithm's optimality assumptions and was therefore not adopted in subsequent experiments.

Table 5-1 summarizes the performance on the Oval-track at a reference speed of 1.5 m/s. Baseline MPPI achieves low mean lateral error but does so at the cost of very high steering rate activity. This indicates multiple high-frequency steering corrections that manifest as oscillatory vehicle behavior.

**Table 5-1:** Laboratory performance on the Oval track at $v_{\text{ref}} = 1.5, \text{m/s}$.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10}$ cm [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ | $\lvert\beta\rvert$ [deg] |
|---|---|---|---|---|---|
| MPPI | 1.53 | **0.035** | 0.993 | **38.34** | 1.769 |
| MPPI ($\lambda = 0.5$) | 1.46 | 0.066 | 0.834 | 6.88 | 0.863 |
| MPPI ($\sigma_t = 0.05,\ \sigma_s = 0.1$) | 1.49 | 0.053 | 0.910 | 12.90 | 0.920 |
| MPPI + filter | 1.52 | 0.063 | 0.824 | 6.12 | 0.827 |

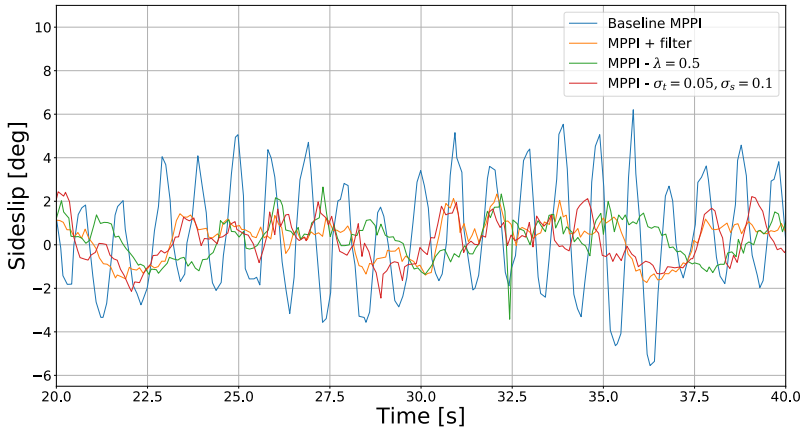



**Figure 5-4:** Side-slip angle $\beta$ for different MPPI variants under laboratory conditions.

In contrast, all three smoothing methods substantially reduces RMS steering rate. Although these modifications do not consistently minimize lateral error or maintain the 10 cm bound,

they yield significantly more stable trajectories. This stability is reflected in the side-slip angle $\beta$, a representation of which is shown in Figure 5-4, where baseline MPPI clearly exhibits the largest oscillation amplitudes.

At higher reference speeds, steering smoothness becomes increasingly critical. Table 5-2 presents results on the Oval-track at 2.5 m/s. While baseline MPPI maintains competitive mean speed, it again exhibits extremely high steering rate activity, resulting in frequent oscillations and degraded tracking metrics. In this regime, the smoothed variants no longer merely improve qualitative behavior but also become competitive in lateral tracking performance, as the cost of oscillations increasingly starts to dominate the stability of the vehicle.

**Table 5-2:** Laboratory performance on the Oval track at $v_{\text{ref}} = 2.5, \text{m/s}$.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10}$ cm [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| MPPI | 2.37 | 0.125 | 0.574 | 96.73 |
| MPPI ($\lambda = 0.5$) | 2.42 | **0.070** | **0.855** | 17.81 |
| MPPI ($\sigma_t = 0.05,\ \sigma_s = 0.1$) | 2.52 | 0.084 | 0.776 | 29.32 |
| MPPI + filter | 2.49 | 0.135 | 0.647 | **8.89** |

Since these methods increase the smoothness of both control inputs (steering and throttle), to isolate whether the observed oscillations originate from longitudinal control, additional experiments were conducted with the throttle command held constant. The oscillatory behavior persisted unchanged, confirming that throttle variability is not the primary driver. This further supports the conclusion that excessive steering rate, rather than longitudinal control dynamics, dominates the observed instability.

### 5-2-2 Frequency Characterization

Section 4-2-2 showed that in simulation an oscillatory mode can be excited by steering-channel phase lag, evidenced by a post-corner spectral peak in the steering command. The following analysis complements the frequency analysis of these oscillations done in Section 4-3 by identifying whether the laboratory oscillations occur at a repeatable characteristic rate. On the Sharp-Corner track at $v_{\text{ref}} = 1.5$ m/s, the vehicle exhibits repeated oscillations about the reference trajectory in the post-corner segment. Figure 5-5 shows a representative run, illustrating the corresponding oscillatory behavior in the side-slip angle.

Analysis across multiple runs shows that the post-corner oscillations are not purely stochastic, but exhibit a repeatable structure. In particular, the post-corner segment consistently shows elevated oscillatory activity with slow decay in the time domain. To characterize this behavior more precisely, a frequency-domain analysis is performed on the published steering command using fixed windows before and after the corner. Figure 5-6 illustrates this analysis for a representative run. Across all runs, a pronounced spectral component around approximately 1 Hz appears in the post-corner window, while this component is substantially weaker before the corner.

The same characteristic oscillation is not limited to sharp-corner geometries. For less severe corners driven at higher reference speed, sustained steering oscillations are observed throughout the corner rather than being confined to corner exit. Importantly, the dominant oscillation

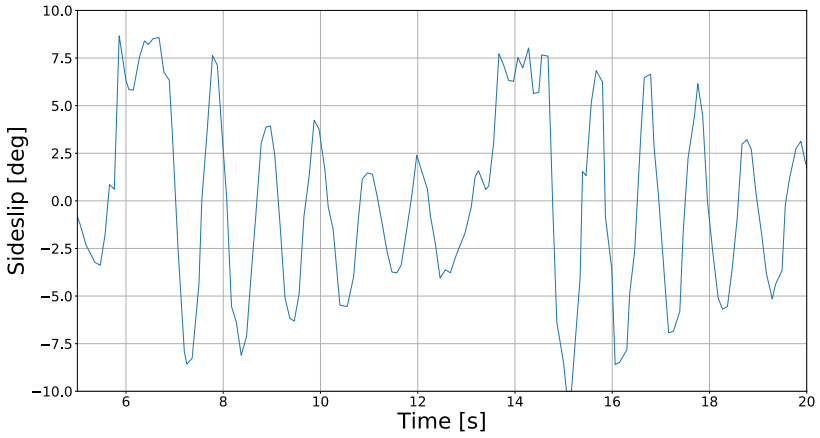
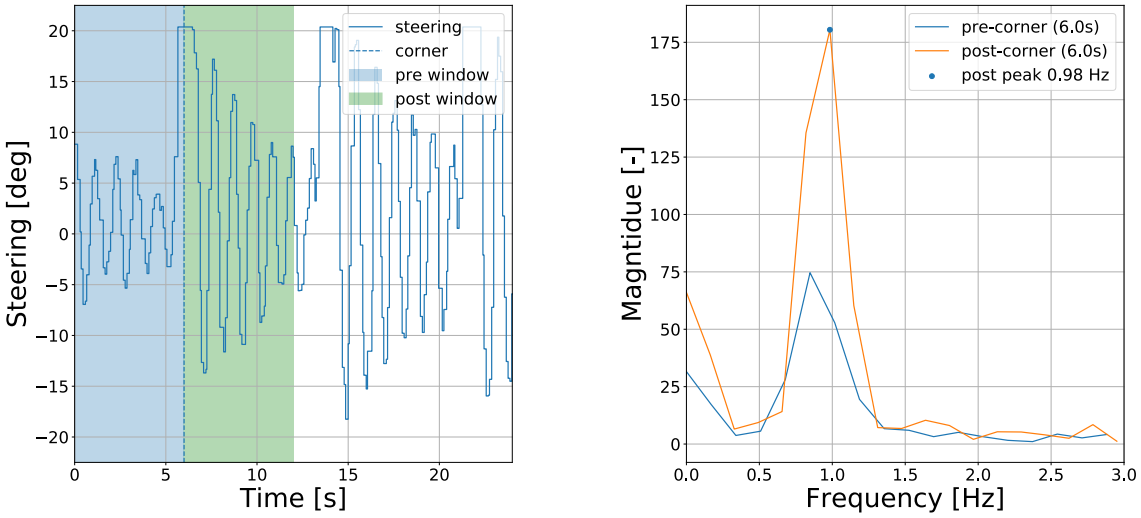



**Figure 5-5:** Time-domain side-slip angle response for a representative laboratory run on the Sharp-Corner track at $v_{\text{ref}} = 1.5, \text{m/s}$, illustrating post-corner oscillations.



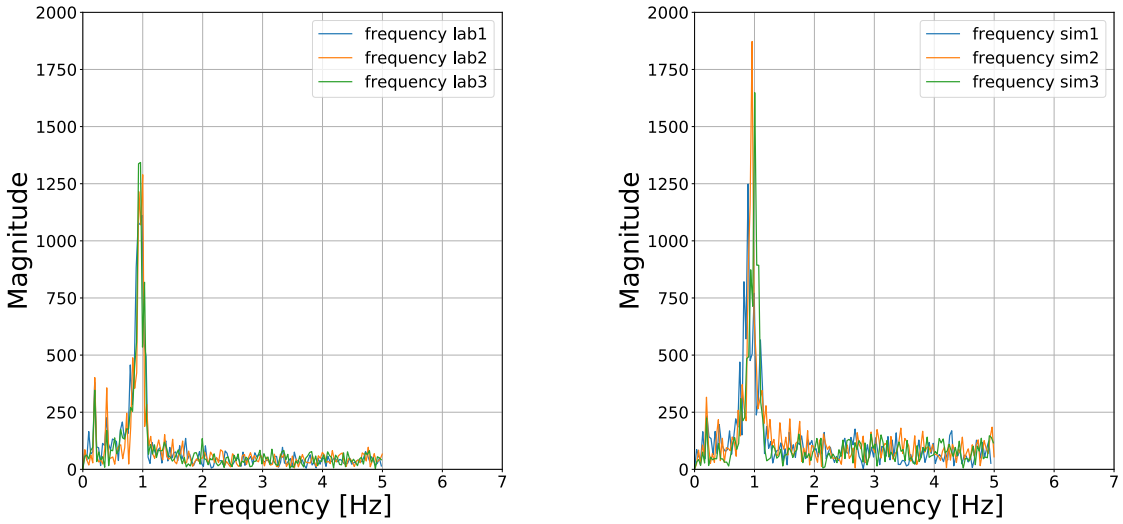

**(a)** Steering command in the time domain, with the pre- and post-corner analysis windows indicated.

**(b)** Steering command spectra computed over fixed windows before and after the sharp corner.

**Figure 5-6:** Frequency-domain characterization of post-corner steering oscillations on the Sharp-Corner track for a representative laboratory run.

frequency remains close to 1 Hz, indicating that this behavior reflects a general closed-loop phenomenon rather than a geometry-specific effect. This is illustrated in Figure 5-7, which shows steering command spectra for three repeated runs (each comprised of multiple laps) on the Oval-track at $v_{\text{ref}} = 2.5$ m/s compared with the same track and a 0.1 s delay simulation. Across all runs, a consistent spectral component near 1 Hz is observed, demonstrating that the oscillations are repeatable and persist during steady-state cornering. Comparison with

the delay-injected simulation reveals similar frequency peaks. Although the frequency-domain agreement is not exact, likely due to the simplified delay model used in simulation, the qualitative correspondence is clear. Indicating that steering-channel lag plays a dominant role in shaping the observed oscillatory behavior.




**(a)** FFT plot of three runs on the Oval-track in the laboratory.

**(b)** FFT plot of three runs on the Oval-track in simulation.

**Figure 5-7:** Frequency-domain characterization of steering oscillations on the Oval track at $v_{\text{ref}} = 2.5, \text{m/s}$, comparing multiple laboratory runs and delay-injected simulation.

The consistent emergence of a dominant component near 1 Hz across different geometries and speeds indicates that the observed oscillations are not incidental, but reflect excitation of inherent closed-loop vehicle dynamics. This frequency range coincides with typical yaw-response bandwidth in road vehicles, where steering inputs couple strongly into lateral and yaw motion [52]. High steering-rate activity is therefore particularly prone to amplifying lateral oscillations [2]. Moreover, high-frequency steering excitation is known to amplify modeling inaccuracies and degrade sim-to-real transfer performance [11]. It has also been established that delay-induced oscillations can be mitigated by reducing effective feedback aggressiveness, thereby trading responsiveness for improved robustness [16].

Together with the time-domain laboratory results and the controller-level intervention study, a consistent interpretation emerges. Oscillatory behavior intensifies with increased steering-rate activity, persists at a repeatable frequency near 1 Hz, and is similar to simulated oscillatory behavior when steering-channel delay is introduced. Interventions that reduce steering-rate content—through increased temperature, reduced sampling covariance, or filtering—systematically suppress this oscillatory mode without altering the tracking objective. The evidence therefore supports the conclusion that the dominant laboratory oscillations arise from the interaction between excessive steering-rate excitation and steering-channel lag, rather than from a fundamental limitation of MPPI itself. For stable laboratory operation, steering-rate smoothing is thus not merely advantageous but practically necessary.

## 5-3 Extension Comparison

Having identified steering rate activity and steering-channel lag as the primary drivers of oscillatory behavior, the proposed MPPI extensions are now evaluated systematically in laboratory experiments. The objective of this section is to assess how the different smoothness-oriented modifications affect the trade-off between tracking performance, robustness, and responsiveness. All extensions are evaluated on multiple track configurations, reference speeds and scenario's.

### 5-3-1 Oval-track

On the Oval-track at $v_{\text{ref}} = 1.5$ m/s, baseline MPPI achieves the best tracking metrics, Table 5-3. At this low speed, oscillations are present but do not yet dominate the performance metrics. The extensions reduce steering activity substantially, but Dynamic Covariance MPPI and Smooth MPPI (SMPPI) show degraded lateral tracking, indicating that excessive smoothing can worsen performance when accurate tracking requires fast steering corrections.

**Table 5-3:** Performance on the Oval-track in the laboratory at $v_{\text{ref}} = 1.5$ m/s.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | TIB$<$ 10 cm [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| MPPI | 1.53 | **0.035** | **0.998** | 38.34 |
| Dyn-cov MPPI | 1.56 | 0.074 | 0.747 | 9.24 |
| SMPPI | 1.61 | 0.119 | 0.550 | 7.55 |
| LFS-MPPI | 1.54 | 0.045 | 0.997 | 13.67 |

Table 5-4 summarizes the behavior on the Oval-track at $v_{\text{ref}} = 2.5$ m/s. Baseline MPPI achieves competitive mean speed but exhibits the highest steering activity by a large margin, which coincides with the lowest Time-In-Bound (TIB) and the largest RMS lateral error. This indicates that at higher speeds, steering smoothness directly influences tracking performance, just as was seen in the simulation. Dynamic covariance MPPI yields the lowest RMS steering rate in this comparison, but the improvement in smoothness does not translate into robust tracking. Although TIB improves relative to baseline, lateral error remains high. These results indicate that adaptive covariance scaling can suppress steering activity, but may also reduce the exploration needed to recover from disturbances or prediction error.

**Table 5-4:** Performance of extensions on the Oval-track at $v_{\text{ref}} = 2.5$ m/s.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | TIB$_{<10}$ cm [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| MPPI | 2.37 | 0.125 | 0.574 | 96.73 |
| Dyn-cov MPPI | 2.48 | 0.092 | 0.713 | **16.29** |
| SMPPI | **2.51** | 0.114 | 0.656 | 26.48 |
| LFS-MPPI | 2.46 | **0.071** | **0.840** | 42.23 |

SMPPI similarly reduces steering activity relative to baseline while maintaining the highest mean speed in this comparison. However, its lateral tracking performance remains limited, consistent with the reduced bandwidth and saturation-induced loss of sampling efficiency discussed. In demanding segments requiring rapid steering transients, the rate-based parametrization suppresses oscillations but can also hinder aggressive corrective action.

Low-pass Filtered Sampling MPPI (LFS-MPPI) provides the most favorable overall trade-off. Compared to baseline MPPI, it substantially improves tracking and robustness metrics, while maintaining a comparable mean speed. Although its RMS steering rate is not the lowest in Table 5-4, it is reduced sufficiently to suppress oscillations while preserving the controller's ability to execute sustained, high-curvature maneuvers.

### 5-3-2 Sharp-Corner track

Additional insight is obtained by evaluating post-corner settling behavior on the Sharp-Corner track. To isolate this the Sharp-Corner track is used and the settling times are considered at $v_{\text{ref}} = 1.5\,\text{m/s}$ and grouped in Table 5-5. Only the side-slip angle settling time is reported, as other performance metrics mirror those observed on the Oval-track. In this table MPPI often didn't even settle before the next corner came up. The three extensions that are evaluated are very close to each other regarding settling time, however these still are related to the steering rates seen in the previous table, where the extensions go up in settling time in the same order as they go up from the RMS steering rate.

**Table 5-5:** Side-slip angle settling time on the Sharp-Corner track at $v_{\text{ref}} = 1.5, \text{m/s}$.

| Controller | settling time $\beta$ [s] |
|---|---|
| MPPI | 5.090 |
| Dyn-cov MPPI | 1.957 |
| SMPPI | 2.060 |
| LFS-MPPI | 2.118 |

### 5-3-3 Obstacle Avoidance

A static obstacle is introduced in the laboratory by augmenting the running cost with a soft penalty identical to the simulation setup. The obstacle is modeled as an ellipse placed on the reference trajectory on the Oval-track at $v_{\text{ref}} = 1.5$ m/s, with $a = 0.50$ m and $b = 0.25$ m.

Table 5-6 summarizes the resulting performance. All controllers maintain a minimum clearance exceeding 0.20 m with respect to the obstacle center, thereby avoiding physical collision (DART has a conservative physical radius of about 0.15 m). However, differences in avoidance strategy are clearly reflected in steering activity and envelope clearance.

**Table 5-6:** Laboratory performance on the Oval track with static obstacle avoidance at $v_{\text{ref}} = 1.5, \text{m/s}$.

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ | $d_{\text{E,min}}$ [m] | $d_{\text{C,min}}$ [m] |
|---|---|---|---|---|
| MPPI | 1.40 | 69.18 | -0.062 | 0.213 |
| Dyn-cov MPPI | **0.86** | 4.25 | 0.064 | **0.622** |
| SMPPI | 1.40 | **31.15** | -0.056 | 0.248 |
| LFS-MPPI | 1.50 | **37.92** | -0.092 | 0.230 |

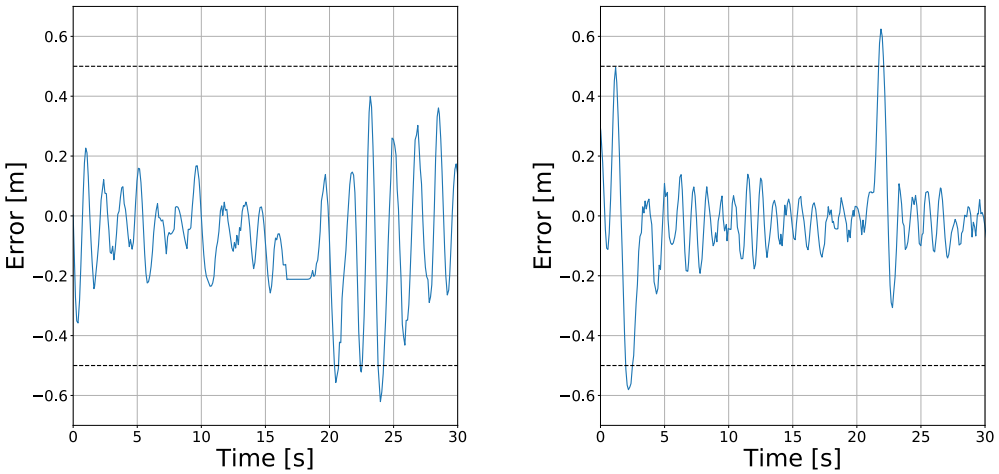
For Dynamic Covariance MPPI, the low mean speed reflects a failure mode in which the vehicle slows down and eventually stops behind the obstacle. In this regime, the contracted

steering covariance limits exploration of sufficiently large steering commands to initiate an evasive maneuver, resulting in a "stuck" behavior. In contrast, baseline MPPI, SMPPI, and LFS-MPPI exhibit more aggressive steering, which reduces clearance and leads to occasional penetration of the inflated safety envelope. This sensitivity is consistent with unmodeled real-world effects (tracking error, discretization, and actuator dynamics) that can turn near-envelope maneuvers into envelope violations.

**Table 5-7:** Laboratory performance on the Oval track with dynamic obstacle avoidance at nominal temperature.

| Controller | $\bar{v}$ $[\frac{\mathrm{m}}{\mathrm{s}}]$ | $\dot{\delta}_{\mathrm{RMS}}[\frac{\mathrm{deg}}{\mathrm{s}}]$ | $\mathrm{TIB}_{<50\mathrm{cm}}$ | $d_{\mathrm{E,min}}$ [m] | $d_{\mathrm{C,min}}$ [m] |
|---|---|---|---|---|---|
| MPPI | 2.07 | 78.41 | 1.00 | -0.155 | 0.184 |
| Dyn-cov MPPI | 2.25 | 32.75 | 1.00 | -0.135 | 0.229 |
| SMPPI | 2.17 | **81.05** | 0.97 | -0.101 | 0.222 |
| LFS-MPPI | 2.26 | 65.77 | 0.97 | -0.187 | **0.081** |

A second experiment considers a moving obstacle in the laboratory. The obstacle, with $a = 0.40$ m, $b = 0.15$ m, follows a predefined trajectory along the Oval-track at $v = 1.0$ m/s, and the safety envelope evolves over the prediction horizon analogously to the simulation setup. The dynamic obstacle experiments, summarized in Table 5-7, reveal a strong sensitivity to controller aggressiveness.




**(a)** SMPPI. **(b)** LFS-MPPI.

**Figure 5-8:** Lateral error over time for SMPPI and LFS-MPPI showing that they go out of the lane, with the lanes shown by the dotted lines at the $0.5\,\mathrm{m}$ lateral error lines.

At nominal temperature, baseline MPPI and SMPPI produce the highest steering activity, reflected in large $\dot{\delta}_{\mathrm{RMS}}$. These high rates originate from abrupt avoidance maneuvers when the vehicle approaches the envelope, leading to large steering spikes and, in several runs, unstable post-avoidance transients. LFS-MPPI also leaves the lane in some runs, but typically re-enters quickly and stabilizes, whereas SMPPI can exhibit prolonged oscillations after an overtake attempt, visualized in Figure 5-8.

Across controllers, mean speeds lie below $v_{\text{ref}}$ because the vehicle often initially follows the moving obstacle at reduced speed before a sufficient number of low-cost rollouts that pass the obstacle become available. Once these rollouts dominate the importance weights, the mean control sequence shifts toward an overtaking maneuver and the pass is initiated.

Increasing the temperature has a stabilizing effect. As shown in Table 5-8, steering activity decreases substantially for all controllers while mean speed and TIB remain comparable. In particular, MPPI and LFS-MPPI benefit strongly, indicating that reduced sampling aggressiveness mitigates the sharp corrective actions that arise during moving-obstacle avoidance. Dynamic Covariance MPPI remains the smoothest variant in both settings. In contrast, SMPPI continues to exhibit poor avoidance behavior at higher temperature, reflected by reduced clearance and lane-exit events.

**Table 5-8:** Laboratory performance on the Oval track with dynamic obstacle avoidance at increased temperature ($\lambda = 0.5, \lambda_{\text{baseline}}$).

| Controller | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ | $\text{TIB}_{<50\text{cm}}$ | $d_{\text{E,min}}$ [m] | $d_{\text{C,min}}$ [m] |
|---|---|---|---|---|---|
| MPPI | 2.29 | 34.02 | 1.00 | -0.084 | **0.314** |
| Dyn-cov MPPI | 2.20 | **6.21** | 1.00 | -0.045 | 0.266 |
| SMPPI | 2.33 | 37.99 | 0.96 | -0.136 | 0.114 |
| LFS-MPPI | 2.34 | 18.31 | **1.00** | -0.063 | **0.311** |

### 5-3-4 Racetrack

Because baseline MPPI and SMPPI exhibited unsafe high-amplitude oscillations at $v_{\text{ref}} = 2.5$ m/s on the Oval-track, only Dynamic Covariance MPPI and LFS-MPPI were evaluated further on the Racetrack with high reference speed.

For Dynamic Covariance MPPI, failure cases were observed in the sharp-corner scenarios, where the vehicle becomes effectively "stuck", as illustrated in Figure 5-9. This behavior is consistent with the covariance contraction mechanism observed in simulation, but manifests more severely on the physical platform. In the real system, excessive reduction of the sampling covariance can suppress steering exploration to such an extent that recovery from large lateral deviations becomes impossible.

At lower reference speeds, this failure mode does not occur and the controller is able to negotiate the sharp corners successfully, indicating that the issue arises primarily under high-demand conditions. The dynamic covariance parameters were retuned iteratively, both in simulation and on hardware, but this did not eliminate the failure cases. Notably, similar behavior was previously observed in the static obstacle avoidance scenario, suggesting that this sensitivity to covariance contraction represents a structural limitation of the adaptive scheme rather than a scenario-specific tuning deficiency.

LFS-MPPI enables stable high-speed driving even with the kinematic prediction model. Residual oscillations can still occur near the performance limit, but they are mitigated by combining LFS-MPPI with a moderately increased temperature. Figure 5-10 shows the resulting Racetrack trajectory at $v_{\text{ref}} = 2.5$ m/s, illustrating stable tracking without the pronounced oscillatory steering observed for baseline MPPI at increased temperature.

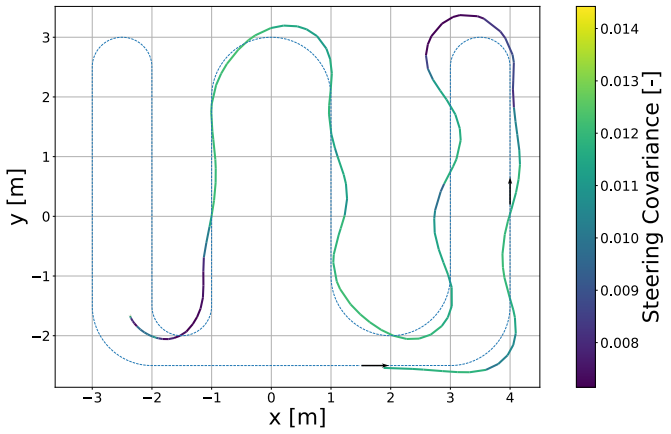



**Figure 5-9:** Spatial evolution of steering covariance $\sigma_s$ along the laboratory-driven path at $v_{\text{ref}} = 2.5, \text{m/s}$.

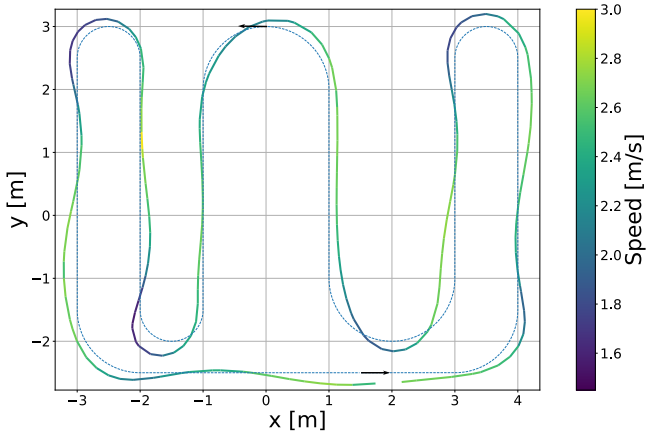



**Figure 5-10:** Laboratory trajectory on the Racetrack at $v_{\text{ref}} = 2.5, \text{m/s}$ for LFS-MPPI.

## 5-4 Dynamic Prediction Model

In laboratory experiments, the sampling–fidelity trade-off becomes more pronounced because real-time feasibility is strictly enforced. Unlike simulation, where the plant can pause while the controller computes a control update, the physical platform requires that each control sequence be computed within the fixed $dt = 0.1\,\text{s}$ period. Consequently, reductions in sample count directly affect closed-loop stability.

On the Oval-track at $v_{\text{ref}} = 1.5$ m/s, the effect of sample reduction is evident. With $J = 500$, the vehicle exhibits unstable steering oscillations. Increasing the sample count to $J = 1000$ and $J = 2000$ improves stability, while $J = 4000$ leads to frequent computational saturation, causing intermittent delays in control updates and degraded performance. The corresponding performance metrics are summarized in Table 5-9. These results demonstrate that both

insufficient sampling ($J$ too small) and computational overload ($J$ too large) can degrade hardware performance.

**Table 5-9:** Laboratory performance on the Oval track at $v_{\text{ref}} = 1.5, \text{m/s}$ as a function of rollout count $J$.

| $J$ | $\bar{v}$ $[\frac{\text{m}}{\text{s}}]$ | $e_{\text{lat,RMS}}$ [m] | $\text{TIB}_{<10\text{ cm}}$ [-] | $\dot{\delta}_{\text{RMS}}[\frac{\text{deg}}{\text{s}}]$ |
|---|---|---|---|---|
| 500 | 1.51 | 0.059 | 0.918 | **70.32** |
| 1000 | 1.53 | 0.043 | 0.922 | 50.73 |
| 2000 | 1.55 | 0.038 | **0.991** | 40.23 |
| 4000 | 1.49 | **0.138** | 0.803 | 49.08 |

Enabling low-pass filtered sampling restores stable operation at $J = 500$, confirming that improved sampling structure can compensate for reduced rollout count. The advantage of the dynamic prediction model becomes particularly apparent at higher reference speeds, where nonlinear tire dynamics dominate and the kinematic model begins to lose predictive accuracy. On the Racetrack at $v_{\text{ref}} = 3.0$ m/s, the kinematic prediction model completes only approximately 50% of runs. In these fail cases, the vehicle either goes to wide out of track crashing into the sides of the laboratory or spins around from which recovery is not achieved. In contrast, the dynamic prediction model maintains closed-loop stability under the same conditions. Representative runs are shown in Figure 5-11.

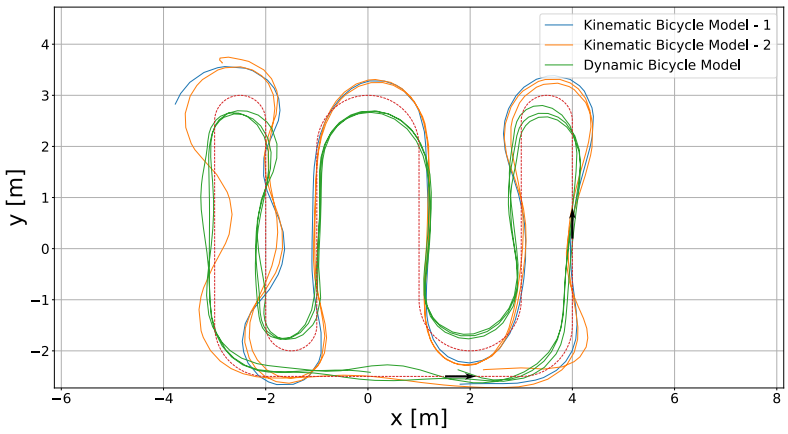



**Figure 5-11:** Laboratory trajectories on the Racetrack at $v_{\text{ref}} = 3.0, \text{m/s}$ for LFS-MPPI with kinematic and dynamic prediction models.

With the dynamic model and LFS-MPPI, stable tracking on the Racetrack is achieved up to 3.5 m/s, as shown in Figure 5-12. This improvement arises from combining higher-fidelity prediction with structured sampling that preserves effective rollout quality under real-time constraints. Together, these results demonstrate that smoothness-oriented sampling enables deployment of higher-fidelity predictive models near the handling limit without exceeding computational budgets.

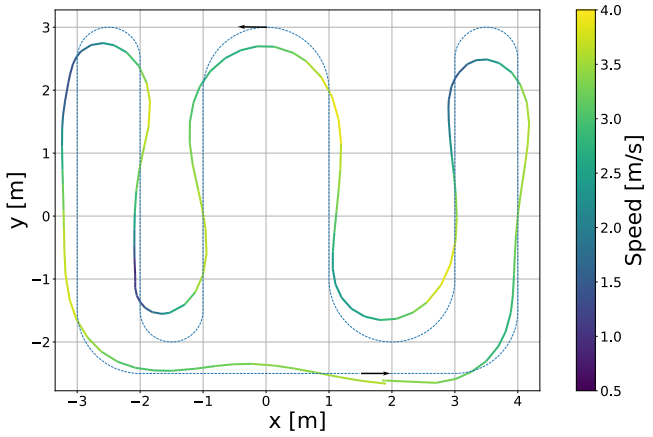



**Figure 5-12:** Laboratory trajectory on the Racetrack at $v_{\text{ref}} = 3.5, \text{m/s}$ for LFS-MPPI with the dynamic prediction model.

# Chapter 6

# Conclusion

This thesis investigated the practical deployment of sampling-based control for high-speed autonomous driving, focusing on Model Predictive Path Integral (MPPI) and smoothness-oriented extensions. The objective was to characterize sim-to-real behavior on a small-scale racing platform and to identify algorithmic modifications that mitigate oscillatory behavior while preserving tracking performance near the handling limit.

Baseline MPPI achieves accurate tracking and competitive speeds in both simulation and laboratory experiments. However, its sampling-based update generates significant high-frequency steering content, which induces yaw-rate and side-slip oscillations in the laboratory experiments. These oscillations are amplified by actuator delay and limited steering bandwidth and constitute the dominant real-world performance limitation.

A comparison with Model Predictive Control (MPC) clarifies the relative strengths of sampling-based control. MPC offers superior nominal tracking under accurate modeling and strict feasibility, whereas MPPI degrades more gracefully under model mismatch and avoids hard feasibility failures, albeit with higher control variability.

Among the evaluated extensions, Low-pass Filtered Sampling MPPI (LFS-MPPI) provides the most consistent trade-off between tracking accuracy, smoothness, and robustness. While Dynamic Covariance MPPI and Smooth MPPI suppress oscillations, they can overly restrict exploration or reduce effective control bandwidth during aggressive maneuvers. In contrast, LFS-MPPI preserves exploration while selectively attenuating unnecessary high-frequency excitation.

A central insight is that model fidelity and sampling efficiency are intrinsically coupled under real-time constraints. Increasing prediction-model fidelity reduces the feasible rollout count, which increases control variance and can negate the benefits of improved modeling. Temporally structured sampling alters this trade-off. In particular, LFS-MPPI increases the proportion of dynamically plausible rollouts without increasing computational cost, enabling stable deployment of the dynamic model at reduced sample counts. In laboratory experiments, this allowed stable operation at $J = 500$ and reliable tracking up to $3.5\,\mathrm{m/s}$.

Taken together, three conclusions emerge. First, baseline MPPI requires explicit steering rate moderation for stable hardware deployment. Second, temporally structured sampling provides a principled way to improve robustness without sacrificing tracking performance. Third, structured sampling enables high-fidelity predictive control within strict computational budgets, expanding the feasible operating envelope of sampling-based methods.

This thesis bridges the gap between simulation-focused MPPI studies and real-world deployment by demonstrating how sampling structure, model fidelity, and computational limits interact in practice. It provides concrete guidelines for applying sampling-based predictive control in safety-critical, high-performance autonomous systems.

Future work may investigate combining temporally structured sampling with adaptive temperature or covariance schemes, incorporating explicit actuator-delay models within the prediction dynamics, and extending validation to multi-agent and more complex dynamic environments.

## 6-1 Answers to Research Questions

**How does baseline MPPI perform when deployed on a real-world small-scale autonomous racing platform operating near the limits of handling?**

- Baseline MPPI achieves reference speeds and low lateral error on the real-world platform, but exhibits pronounced oscillatory behavior, especially when operating near the handling limits. In particular, high-frequency steering commands induce yaw-rate and side-slip angle oscillations that degrade closed-loop stability in the presence of actuator dynamics and delay. While these effects are absent in simulation, they become a dominant performance-limiting factor on physical hardware.

**Which implementation choices and modifications are required to achieve stable and reliable baseline MPPI performance on physical hardware?**

- Stable real-world MPPI deployment requires explicit control over steering rate aggressiveness rather than tuning tracking objectives alone. Increasing temperature, reducing sampling covariance, or introducing temporal correlation in the sampled perturbations all improve stability by attenuating high-frequency steering content. Unlike temperature or covariance reduction, which globally dampen exploration, temporally structured sampling selectively suppresses high-frequency components while preserving low-frequency corrective authority.

**To what extent do improvements observed in simulation translate to real-world performance under identical experimental conditions?**

- While several trends observed in simulation, such as the relationship between sampling aggressiveness and tracking accuracy, carry over qualitatively, real-world experiments reveal additional failure modes that are not captured in nominal simulation. In particular, actuator delay and limited steering bandwidth strongly amplify high-frequency

control actions, leading to instabilities that are absent in simulation. As a result, real-world experiments expose oscillations and failure cases in which the vehicle is unable to recover after aggressive cornering maneuvers or becomes effectively "stuck" on the track. Consequently, while simulation provides reliable qualitative guidance, quantitative stability margins must be validated experimentally.

**What trade-offs arise between control smoothness, responsiveness, and tracking accuracy in real-world MPPI-based control?**

- The results reveal a fundamental trade-off between suppressing high-frequency steering actions and retaining sufficient responsiveness for aggressive trajectory corrections. Excessive smoothing improves stability but can hinder recovery in sharp corners, while aggressive sampling improves tracking at the cost of oscillatory behavior. Effective real-world MPPI deployment therefore requires shaping control smoothness at the level of the sampling process, enabling selective aggressiveness while avoiding unnecessary high-frequency excitation.

**How do smoothness-oriented MPPI extensions affect tracking performance, control stability, and robustness in real-world high-speed racing scenarios?**

- Smoothness-oriented extensions consistently reduce oscillatory steering behavior but differ in how they trade smoothness against responsiveness. Dynamic covariance MPPI and Smooth MPPI effectively suppress steering rate bursts but tend to over-constrain exploration or reduce bandwidth, leading to degraded tracking in aggressive maneuvers. LFS-MPPI provides the most favorable balance, reducing oscillations while maintaining accurate tracking and robustness under model mismatch and actuator delay.
- By improving sampling efficiency through temporally correlated perturbations, LFS-MPPI enables a reduction in rollout count without sacrificing closed-loop stability. Under a fixed computational budget, this makes it feasible to deploy the more accurate dynamic bicycle model as the prediction model in real time. The resulting increase in model fidelity improves predictive consistency at higher lateral accelerations, thereby supporting higher reference speeds and expanding the feasible operating envelope within the tested experimental conditions.

# Bibliography


[1] J3016_202104: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles - SAE International.

[2] M. Abe and W. Manning. Chapter 3 - fundamentals of vehicle dynamics. In M. Abe and W. Manning, editors, *Vehicle Handling Dynamics*, pages 47–118. Butterworth-Heinemann, Oxford, 2009.

[3] Masato Abe. Fundamentals of Vehicle Dynamics. In *Vehicle Handling Dynamics*, pages 45–107. Elsevier, 2015.

[4] Elie Aljalbout, Jiaxu Xing, Angel Romero, Iretiayo Akinola, Caelan Reed Garrett, Eric Heiden, Abhishek Gupta, Tucker Hermans, Yashraj Narang, Dieter Fox, et al. The reality gap in robotics: Challenges, solutions, and best practices. *Annual Review of Control, Robotics, and Autonomous Systems*, 9, 2025.

[5] Joel A. E. Andersson, Joris Gillis, Greg Horn, James B. Rawlings, and Moritz Diehl. CasADi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, March 2019.

[6] Dylan M Asmar, Ransalu Senanayake, Shawn Manuel, and Mykel J Kochenderfer. Model predictive optimized path integral strategies. *arXiv preprint arXiv:2203.16633*, 2022.

[7] Karl Johan Åström and Richard Murray. *Feedback systems: an introduction for scientists and engineers*. Princeton university press, 2021.

[8] Johannes Betz, Hongrui Zheng, Alexander Liniger, Ugo Rosolia, Phillip Karle, Madhur Behl, Venkat Krovi, and Rahul Mangharam. Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing. *IEEE Open Journal of Intelligent Transportation Systems*, 3:458–488, 2022.

[9] Mattia Boggio, Carlo Novara, and Michele Taragna. Trajectory planning and control for autonomous vehicles: a "fast" data-aided NMPC approach. *European Journal of Control*, 74:100857, November 2023.

[10] Javier Borquez, Luke Raus, Yusuf Umut Ciftci, and Somil Bansal. DualGuard MPPI: Safe and Performant Optimal Control by Combining Sampling-Based MPC and Hamilton-Jacobi Reachability, February 2025.

[11] Eugenio Chisari, Alexander Liniger, Alisa Rupenyan, Luc Van Gool, and John Lygeros. Learning from simulation, racing in reality. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8046–8052. IEEE, 2021.

[12] Alexander Domahidi and Juan Jerez. Forces professional. Embotech AG, https://embotech.com/FORCES-Pro, 2014–2019. Accessed: February 2025.

[13] Ádám Domina and Viktor Tihanyi. LTV-MPC Approach for Automated Vehicle Path Following at the Limit of Handling. *Sensors*, 22(15):5807, August 2022.

[14] Odichimnma Ezeji, Michael Ziegltrum, Giulio Turrisi, Tommaso Belvedere, and Valerio Modugno. Bc-mppi: A probabilistic constraint layer for safe model-predictive path-integral control. In *Workshop on Agents and Robots for reliable Engineered Autonomy*, pages 131–143. Springer, 2025.

[15] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Hongtei Eric Tseng, and Davor Hrovat. Predictive Active Steering Control for Autonomous Vehicle Systems. *IEEE Transactions on Control Systems Technology*, 15(3):566–580, May 2007.

[16] Karin Festl and Michael Stolz. A nonlinear dead-time compensation method for path tracking control. *arXiv preprint arXiv:2507.06935*, 2025.

[17] Aman Goel, Thomas Detlefsen, and Brian Park. Mppi for robust & efficient racing.

[18] Brian Goldfain, Paul Drews, Changxi You, Matthew Barulic, Orlin Velev, Panagiotis Tsiotras, and James M Rehg. Autorally: An open platform for aggressive autonomous driving. *IEEE Control Systems Magazine*, 39(1):26–55, 2019.

[19] Lars Grüne and Jürgen Pannek. *Nonlinear Model Predictive Control*. Communications and Control Engineering. Springer International Publishing, Cham, 2017.

[20] Bernd Heissing and Metin Ersoy, editors. *Chassis handbook: fundamentals, driving dynamics, components, mechatronics, perspectives*. ATZ. Vieweg + Teubner, Wiesbaden, 1st edition edition, 2011.

[21] Matanya B Horowitz, Anil Damle, and Joel W Burdick. Linear hamilton jacobi bellman equations in high dimensions. In *53rd IEEE Conference on Decision and Control*, pages 5880–5887. IEEE, 2014.

[22] Xianjian Jin, Guodong Yin, and Nan Chen. Advanced Estimation Techniques for Vehicle System Dynamic State: A Survey. *Sensors*, 19(19):4289, October 2019.

[23] Muhammad Kazim, JunGee Hong, Min-Gyeom Kim, and Kwang-Ki K. Kim. Recent advances in path integral control for trajectory optimization: An overview in theoretical and algorithmic perspectives. *Annual Reviews in Control*, 57:100931, 2024.

[24] Piotr Kicki. Low-pass sampling in model predictive path integral control. *arXiv preprint arXiv:2503.11717*, 2025.

[25] Taekyung Kim, Gyuhyun Park, Kiho Kwak, Jihwan Bae, and Wonsuk Lee. Smooth Model Predictive Path Integral Control Without Smoothing. *IEEE Robotics and Automation Letters*, 7(4):10406–10413, October 2022.

[26] David Klüner, Simon Schäfer, Lucas Hegerath, Jianye Xu, Julius Kahle, Hazem Ibrahim, Alexandru Kampmann, and Bassam Alrifaee. Simulation to reality: Testbeds and architectures for connected and automated vehicles. *arXiv preprint arXiv:2505.03472*, 2025.

[27] Xiangyiming Kong, Callen Fisher, and Benjamin Evans. A critical evaluation of Pure Pursuit, MPC and MPCC: Balancing simplicity, performance and constraints. *MATEC Web of Conferences*, 406:04013, 2024.

[28] Denise Lam, Chris Manzie, and Malcolm Good. Model predictive contouring control. In *49th IEEE Conference on Decision and Control (CDC)*, pages 6137–6142, Atlanta, GA, USA, December 2010. IEEE.

[29] Minhyeong Lee and Dongjun Lee. Time-correlated model predictive path integral: Smooth action generation for sampling-based control. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14490–14496. IEEE, 2025.

[30] D. Limon, I. Alvarado, T. Alamo, and E.F. Camacho. MPC for tracking piecewise constant references for constrained linear systems. *Automatica*, 44(9):2382–2387, September 2008.

[31] Alexander Liniger, Alexander Domahidi, and Manfred Morari. Optimization-Based Autonomous Racing of 1:43 Scale RC Cars. *Optimal Control Applications and Methods*, 36(5):628–647, September 2015.

[32] J. Lofberg. YALMIP : a toolbox for modeling and optimization in MATLAB. In *2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No.04CH37508)*, pages 284–289, September 2004.

[33] Lorenzo Lyons, Thijs Niesten, and Laura Ferranti. Dart: A compact platform for autonomous driving research. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 129–136. IEEE, 2024.

[34] David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[35] Phong Nguyen. Curse of Dimensionality - Explain with Code. https://www.kaggle.com/code/phongnguyen1/curse-of-dimensionality-explain-with-code, 2023. Accessed: November 2025.

[36] Hans B. Pacejka. Tire Characteristics and Vehicle Handling and Stability. In *Tire and Vehicle Dynamics*, pages 1–58. Elsevier, 2012.

[37] Brian Paden, Michal Cap, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1):33–55, March 2016.

[38] Corrado Pezzato, Chadi Salmi, Max Spahn, Elia Trevisan, Javier Alonso-Mora, and Carlos Hernandez Corbato. Sampling-based model predictive control leveraging parallelizable physics simulations, 2023.

[39] Christoph Rauscher, Volker Janssen, and Roland Minihold. *Fundamentals of spectrum analysis*, volume 25. Rohde & Schwarz Munich, Germany, 2007.

[40] James Blake Rawlings, David Q. Mayne, and Moritz Diehl. *Model predictive control: theory, computation and design*. Nob Hill Publishing, LLC, Santa Barbara, 2nd edition edition, 2020.

[41] P Riekert and T.E. Schunck. Zur Fahrmechanik des gummibereiften Kraftfahrzeugs. *Ing. Arch 11*, 1940.

[42] Mohammad Rokonuzzaman, Navid Mohajer, Saeid Nahavandi, and Shady Mohamed. Review and performance evaluation of path tracking controllers of autonomous vehicles. *IET Intelligent Transport Systems*, 15(5):646–670, 2021.

[43] Abraham Savitzky and Marcel JE Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

[44] Robert Shapiro and Isaac Yoder. Innovation Highway: Unlocking the Social and Economic Benefits of Autonomous Vehicles. 2023.

[45] Yang Shi and Kunwu Zhang. Advanced model predictive control framework for autonomous intelligent mechatronic systems: A tutorial overview and perspectives. *Annual Reviews in Control*, 52:170–196, 2021.

[46] Efstathios Siampis, Efstathios Velenis, Salvatore Gariuolo, and Stefano Longo. A Real-Time Nonlinear Model Predictive Control Strategy for Stabilization of an Electric Vehicle at the Limits of Handling. *IEEE Transactions on Control Systems Technology*, 26(6):1982–1994, November 2018.

[47] Jarrod M Snider. Automatic Steering Methods for Autonomous Automobile Path Tracking.

[48] P. Stano, U. Montanaro, D. Tavernini, M. Tufo, G. Fiengo, L. Novella, and A. Sorniotti. Model predictive path tracking control for automated road vehicles: A review. *Annual Reviews in Control*, 55:194–236, 2023.

[49] Mehdi Testouri, Gamal Elghazaly, and Raphael Frank. Towards a safe real-time motion planning framework for autonomous driving systems: An mppi approach. *arXiv preprint arXiv:2308.01654*, 2023.

[50] Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research*, 11:3137–3181, 2010.

[51] UM-ARM-Lab. pytorch_mppi: Model predictive path integral control in pytorch. https://github.com/UM-ARM-Lab/pytorch_mppi, 2023.

[52] T Van der Sande, P Zegelaar, I Besselink, and H Nijmeijer. A robust control analysis for a steer-by-wire vehicle with uncertainty on the tyre forces. *Vehicle System Dynamics*, 54(9):1247–1268, 2016.

[53] Robin Verschueren, Gianluca Frison, Dimitris Kouzoupis, Jonathan Frey, Niels Van Duijkeren, Andrea Zanelli, Branimir Novoselnik, Thivaharan Albin, Rien Quirynen, and Moritz Diehl. acados—a modular open-source framework for fast embedded optimal control. *Mathematical Programming Computation*, 14(1):147–183, March 2022.

[54] Bogdan Vlahov, Jason Gibson, David D Fan, Patrick Spieler, Ali-akbar Aghamohammadi, and Evangelos A Theodorou. Low frequency sampling in model predictive path integral control. *IEEE Robotics and Automation Letters*, 9(5):4543–4550, 2024.

[55] Fanxin Wang, Yikun Cheng, and Chuyuan Tao. Mppi-dbas: Safe trajectory optimization with adaptive exploration. *arXiv preprint arXiv:2502.14387*, 2025.

[56] Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015.

[57] Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1433–1440. IEEE, 2016.

[58] Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and Evangelos A. Theodorou. Information-Theoretic Model Predictive Control: Theory and Applications to Autonomous Driving. *IEEE Transactions on Robotics*, 34(6):1603–1622, December 2018.

[59] Grady Robert Williams. *Model Predictive Path Integral Control: Theoretical Foundations and Applications to Autonomous Driving*. PhD thesis, Georgia Institute of Technology, March 2019. Ph.D. thesis, School of Computer Science, College of Computing.

[60] Fitri Yakub and Yasuchika Mori. Comparative study of autonomous path-following vehicle control via model predictive control and linear quadratic control. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 229(12):1695–1714, October 2015.

[61] Leon Liangwu Yan and Santosh Devasia. Output-Sampled Model Predictive Path Integral Control (o-MPPI) for Increased Efficiency. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14279–14285, Yokohama, Japan, May 2024. IEEE.

[62] Zeji Yi, Chaoyi Pan, Guanqi He, Guannan Qu, and Guanya Shi. Covo-mpc: Theoretical analysis of sampling-based mpc and optimal covariance design. In *6th Annual Learning for Dynamics & Control Conference*, pages 1122–1135. PMLR, 2024.

[63] Ji Yin, Zhiyuan Zhang, Evangelos Theodorou, and Panagiotis Tsiotras. Trajectory Distribution Control for Model Predictive Path Integral Control using Covariance Steering. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 1478–1484, Philadelphia, PA, USA, May 2022. IEEE.

[64] Jingyuan Zhao, Wenyi Zhao, Bo Deng, Zhenghong Wang, Feng Zhang, Wenxiang Zheng, Wanke Cao, Jinrui Nan, Yubo Lian, and Andrew F. Burke. Autonomous driving system: A comprehensive survey. *Expert Systems with Applications*, 242:122836, May 2024.

# Glossary

## List of Acronyms

| | |
|---|---|
| **AV** | Autonomous Vehicle |
| **MPC** | Model Predictive Control |
| **MPPI** | Model Predictive Path Integral |
| **CG** | center of gravity |
| **RMS** | Root Mean Square |
| **MIMO** | Multiple-Input Multiple-Output |
| **MPCC** | Model Predictive Contouring Control |
| **MPPI** | Model Predictive Path Integral |
| **OCP** | Optimal Control Problem |
| **QP** | Quadratic Program |
| **LTI** | Linear Time-Invariant |
| **LTV** | Linear Time-Varying |
| **LPV** | Linear Parameter-Varying |
| **NMPC** | Nonlinear MPC |
| **PI** | Path Integral |
| **SMPPI** | Smooth MPPI |
| **GPU** | Graphics Processing Unit |
| **HJB** | Hamilton-Jacobi-Bellman |
| **SDE** | Stochastic Differential Equation |
| **PDE** | Partial Differential Equation |
| **KL** | Kullback-Leibler |
| **DART** | Delft's Autonomous-driving Robotic Testbed |
| **TIB** | Time-In-Bound |
| **FD** | Finite-Difference |

**DARPA** Defense Advanced Research Projects Agency

**FFT** Fast Fourier Transform

**LFS-MPPI** Low-pass Filtered Sampling MPPI

## List of Symbols

| | |
|---|---|
| $\beta$ | Side-Slip Angle |
| $\delta$ | Steering Angle |
| $\dot{\delta}_{\text{RMS}}$ | RMS Steering Rate |
| $\epsilon^c$ | Contouring Error |
| $\epsilon^l$ | Lag Error |
| $\eta$ | Normalization Constant |
| $\gamma$ | Rollouts Discount Factor |
| $\Gamma(\cdot)$ | Softplus Function |
| $\lambda$ | Temperature |
| $\nu$ | Control Dimension |
| $\omega$ | Yaw Rate |
| $\phi(\cdot)$ | Terminal Cost Function |
| $\psi$ | Yaw Angle |
| $\Psi(x,t)$ | Desirability Function |
| $\rho$ | Minimum Cost |
| $\Sigma$ | Sampling Covariance Matrix |
| $\sigma_i$ | Sampling Covariance of Input $i$ |
| $\bar{v}$ | Mean Vehicle Speed |
| $\bar{x}$ | State Vector |
| $\hat{F}_x$ | Longitudinal Tire Force |
| $\hat{F}_y$ | Lateral Tire Force |
| $\mathbb{P}$ | Trajectory Distribution Under Uncontrolled Dynamics |
| $\mathbb{Q}$ | Trajectory Distribution Under Controlled Dynamics |
| $B(\cdot)$ | Diffusion |
| $C(\cdot)$ | Constraint Cost Function |
| $c(k)$ | Obstacle Center at Timestep $k$ |
| $d_{\text{C,min}}$ | Minimum Euclidean Distance to Obstacle Center |
| $d_{\text{E,min}}$ | Minimum Distance to Safety Envelope |
| $dw$ | Wiener Process |
| $E(\theta)$ | Planar Rotation Matrix |
| $f(\cdot)$ | Deterministic Dynamic Function |
| $F_f$ | Friction Force |

| | |
|---|---|
| $F_m$ | Motor Force |
| $G(x,t)$ | Control Matrix |
| $I_z$ | Yaw Moment of Inertia |
| $J_{\text{eff}}$ | Effective Sample Size |
| $k$ | Discrete Time Instance |
| $l$ | Wheelbase |
| $L(\cdot)$ | Running Cost Function |
| $l_f, l_r$ | Distances from CG to Front and Rear Axles |
| $m$ | Mass |
| $N$ | Discrete-Time Prediction Horizon |
| $Q(k)$ | Ellipse Matrix at Timestep $k$ |
| $R$ | Control Cost Matrix |
| $S_j$ | Cost-To-Go |
| $T$ | Continuous-Time Prediction Horizon |
| $U$ | Control Sequence |
| $u(k)$ | Control Input at Time $k$ |
| $u^*(k)$ | Computed Control Action |
| $U_s$ | Control Space |
| $V(x,t)$ | Value Function |
| $v_x$ | Longitudinal Velocity |
| $v_y$ | Lateral Velocity |
| $w_j$ | Normalized Importance Weight of Trajectory $j$ |
| $X$ | State Space |
| $x(k)$ | State at Time $k$ |
| $x, y$ | Global Position Coordinates |
| $X_f$ | Terminal Constraint Set |
| $x_r$ | Rear Wheel's Longitudinal Position |
| $x_v, y_v$ | Body Frame Coordinates |
| $y_r$ | Rear Wheel's Lateral Position |