Unifying Model Predictive Path Integral Control: From Stochastic Theory to Real-Time Implementation

Leesai Park[a], Keunwoo Jang[b,*], Sanghyun Kim[a,c,*]

[a] *Department of Mechanical Engineering, Kyung Hee University, Yongin, Republic of Korea*
[b] *Center for Humanoid Research, Korea Institute of Science and Technology (KIST), Seoul, Republic of Korea*
[c] *Advanced Institute of Convergence Technology (AICT), Suwon, Republic of Korea*

Abstract

MPPI (Model Predictive Path Integral) control, which is one of the sampling-based stochastic optimal control methods, has shown significant success in real-time control applications in recent years. However, the rapid proliferation of MPPI variants and extensions has created a fragmented methodological landscape without systematic organization. This survey provides a comprehensive overview of MPPI by first establishing its theoretical foundations through a unified derivation from stochastic optimal control. We then systematically categorize and analyze existing approaches across four key aspects: sampling strategies, constraint handling, framework design, and robustness enhancement. We also review computational implementation strategies for MPPI. Building upon these reviewed implementations, we present a modular, open-source MPPI framework with GPU acceleration, available at `https://github.com/rcilab/rci_mppi_framework`. Finally, we identify several open challenges and future research directions, making this work both a research reference and a technical roadmap for advancing MPPI research.

Keywords: Model Predictive Path Integral, Sampling-based Model Predictive Control, Stochastic Optimal Control, Trajectory Optimization

1. Introduction

Model Predictive Control (MPC) has emerged as one of the most successful optimization-based control frameworks across robotics, autonomous driving, and process control applications [1]. By repeatedly solving finite-horizon optimization problems online, MPC enables systems to anticipate future states and optimize control sequences while explicitly handling constraints such as joint limits, actuator saturation, and collision avoidance. This predictive capability has led to remarkable successes in robotic manipulation [2], legged locomotion [3, 4], and autonomous navigation [5, 6], establishing MPC as a principled alternative to reactive controllers that can systematically encode complex task objectives and safety requirements.

However, traditional MPC approaches rely on gradient-based optimization under the critical assumption that both system dynamics and cost functions remain differentiable [7]. This assumption proves particularly limiting in robotics where systems frequently encounter non-smooth dynamics from contact events, discrete mode switches, and discontinuous cost landscapes. Moreover, the computational burden of derivative calculations scales poorly with system dimensionality, creating a fundamental tension between model fidelity and the stringent real-time requirements of robotic control, which can demand update rates as high as 50–1000 Hz. As a result, these limitations often force researchers and engineers to use simplified models, shorter planning horizons, or conservative approximations preventing MPC from realizing its theoretical advantages in complex robotic systems [8].

To address these limitations, sampling-based optimization techniques have emerged as an attractive alternative, circumventing the need for analytic gradients by evaluating multiple candidate trajectories in parallel. Pioneering works such as Stochastic Trajectory Optimization for Motion Planning (STOMP) [9] and Cross-Entropy Method (CEM) [10, 11] demonstrated the viability of this paradigm, particularly for handling non-differentiable cost functions and leveraging Graphics Processing Unit (GPU) acceleration. However, these early approaches primarily focused on open-loop trajectory optimization rather than closed-loop receding-horizon control, limiting their direct applicability to real-time robotic systems requiring continuous feedback [12].

This gap between sampling-based potential and real-time requirements motivated the development of Model Predictive Path Integral (MPPI) [13] control. MPPI leverages the theoretical foundations of path integral control, which reformulates Stochastic Optimal Control (SOC) as a statistical inference problem, and integrates it with the

[*] Corresponding authors
Email addresses: `leesai2000@khu.ac.kr` (Leesai Park), `jang90@kist.re.kr` (Keunwoo Jang), `kim87@khu.ac.kr` (Sanghyun Kim)

receding-horizon structure of MPC. By expressing the optimal control as a weighted average over sampled noisy trajectories, MPPI sidesteps the need for explicit gradient computation while maintaining theoretical connections to optimal control. This combination of theoretical grounding and algorithmic simplicity, based on sampling, forward simulation, and weighted averaging, has enabled efficient GPU-parallel implementations capable of handling complex, non-smooth cost landscapes found in many real-world control scenarios.

In autonomous ground vehicles, recent developments have demonstrated the effectiveness of MPPI in complex urban environments. Lee *et al.* [14] introduced a GPU-enabled parallel MPPI-based trajectory optimization framework that combines random-sampling exploration with gradient-based refinement, achieving safe motion planning in complex urban driving scenarios. Extending this beyond single-vehicle applications, Matsui *et al.* [15] presented a Prioritised Multi-Agent Model Predictive Path Integral (P-MA-MPPI) method for decentralized coordination of autonomous mining haulage trucks, which reduced deadlocks and collisions while improving energy efficiency by 20%.

In robotic manipulation and vision-based control, MPPI has shown particular promise for reactive, high-frequency control tasks across diverse applications. Bhardwaj *et al.* [16] proposed Stochastic Tensor Optimization for Robot Motion (STORM), a joint-space MPPI-based control framework for reactive manipulation with a robotic arm, achieving 125 Hz real-time control with integrated collision cost learning. Building upon this line of work, Kim *et al.* [17] further advanced high-frequency MPPI-based manipulation by introducing a single-instance sampling strategy and a dynamic time horizon for task-space MPPI control, enabling real-time control at over 1 kHz on a 7-DoF robotic arm while significantly reducing computational burden and maintaining high control accuracy under multiple constraints. Complementing joint-space approaches, Mohamed *et al.* [18] applied an MPPI-based Visual Servoing (MPPI-VS) controller to 6-DoF vision-based robotic control, demonstrating superior constraint handling and robustness compared to classical Image-Based Visual Servoing (IBVS) and Position-Based Visual Servoing (PBVS) schemes. In specialized manipulation tasks, Vasios *et al.* [19] integrated MPPI planning within a Real2Sim2Real pipeline for robotic mushroom harvesting, addressing the unique challenges of dynamic motion planning with deformable objects and soft grippers in agricultural environments.

Aerial and aerospace systems have leveraged the sampling-based nature of MPPI to handle complex flight dynamics and environmental constraints. In quadrotor control, Minavrik *et al.* [20] developed an onboard GPU-parallelized MPPI controller for agile flight in cluttered environments, demonstrating real-time obstacle avoidance at speeds up to 44 km/h. Mohamed *et al.* [21] further explored MPPI for navigation under partial observability, using 3D voxel grid maps to enable safe flight in dynamic, cluttered spaces. Beyond civilian applications, Kim and Lee [22, 23] extended MPPI to missile guidance systems, developing computational guidance algorithms for impact-angle and impact-time control under field-of-view and acceleration constraints.

While these diverse applications demonstrate the versatility and effectiveness of MPPI, the rapid growth of MPPI research has resulted in a fragmented landscape of methodological contributions. Numerous variants have been proposed to address domain-specific challenges, including sampling efficiency enhancement techniques to reduce computational overhead, control barrier functions to enforce safety constraints, hierarchical decompositions for long-horizon planning, and robust formulations to handle model uncertainty, but these advances are scattered across the literature without systematic organization. As a result, the field lacks a clear overarching structure, making it difficult for researchers to gain a coherent understanding of existing approaches, identify common principles, and navigate the rapidly expanding body of work. This fragmentation also hinders the transfer of innovations between domains and complicates the selection of appropriate MPPI variants for new applications.

To unify the scattered approaches and provide crucial structure to the field, this survey offers three key components to the MPPI literature:

- **Theoretical Foundations:** A unified derivation of MPPI from stochastic optimal control and path integral theory, providing the mathematical foundation underlying MPPI and its variants.
- **Methodological Themes:** A systematic classification of MPPI research into four critical themes, namely sampling strategies, constraint handling, framework design, and robustness enhancement, is proposed to provide a coherent structure for the fragmented literature.
- **Implementation Techniques and Open-Source Framework:** An open-source, modular MPPI implementation with GPU-accelerated rollout engines and practical implementation guidelines, serving as both a reference baseline and a development platform for future research in robotics and control.

The remainder of this paper is organized as follows. Section 2 presents the theoretical foundations of MPPI, deriving the path integral formulation from stochastic optimal control and establishing its core algorithmic structure. Section 3 systematically reviews methodological advances in MPPI research across four key themes: sampling strategies, constraint handling, framework design, and robustness enhancement. Section 4 reviews practical developments toward real-time implementation of MPPI and introduces a modular open-source framework with accelerated rollout computation, including baseline experiments that demonstrate the modular design. Section 5 concludes the paper by summarizing the key insights and outlining future research directions for MPPI.

2. Theoretical Foundations of MPPI

This section presents the theoretical foundations of MPPI control through two different derivations within the SOC framework. First, we derive MPPI from an information-theoretic perspective, where trajectory optimization is formulated as minimizing the Kullback–Leibler (KL) divergence between controlled and uncontrolled distributions, yielding the characteristic importance-weighted update rule of MPPI. Second, we present the classical path integral derivation via the Feynman-Kac theorem, transforming the nonlinear Hamilton-Jacobi-Bellman (HJB) equation into a linear PDE whose solution can be expressed as a path integral representation. We demonstrate that both approaches produce identical update laws despite offering distinct interpretations. The information-theoretic view frames MPPI as inference over trajectory distributions, whereas the classical view establishes its connection to dynamic programming. Collectively, these dual perspectives provide a comprehensive theoretical understanding of how MPPI effectively bridges sampling-based exploration with optimal control principles.

2.1. Information-Theoretic Derivation of MPPI

This subsection provides a concise derivation of MPPI from an information-theoretic perspective, following the framework established by Williams et al. [24]. For a more comprehensive treatment of the theoretical details, readers are referred to the original work.

Consider a discrete-time dynamical system

$$\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{v}_t), \qquad \boldsymbol{v}_t \sim \mathcal{N}(\boldsymbol{u}_t, \Sigma), \tag{1}$$

where $\boldsymbol{x}_t \in \mathbb{R}^n$ is the state vector and $\boldsymbol{v}_t \in \mathbb{R}^m$ is a noisy control input drawn from a Gaussian centered at the controllable mean $\boldsymbol{u}_t \in \mathbb{R}^m$. Let $\boldsymbol{U} = (\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{T-1}) \in \mathbb{R}^{m\times T}$ denote the sequence of controllable mean inputs, and $\boldsymbol{V} = (\boldsymbol{v}_0, \ldots, \boldsymbol{v}_{T-1}) \in \mathbb{R}^{m\times T}$ denote the corresponding noisy control sequence.

The probability density of $\boldsymbol{V}$ given $\boldsymbol{U}$ is given by

$$q(\boldsymbol{V}\,|\,\boldsymbol{U},\Sigma) = \prod_{t=0}^{T-1} Z^{-1}\exp\Big(-\tfrac{1}{2}(\boldsymbol{v}_t - \boldsymbol{u}_t)^\top \Sigma^{-1}(\boldsymbol{v}_t - \boldsymbol{u}_t)\Big), \tag{2}$$

where $Z = (2\pi)^{\frac{m}{2}}|\Sigma|^{\frac{1}{2}}$ is the normalization constant of the Gaussian density. We denote by $\mathbb{Q}$ the probability distribution whose density is given by $q(\boldsymbol{V} \mid \boldsymbol{U}, \Sigma)$. Given a control sequence $\boldsymbol{V}$, the resulting state trajectory $\{\boldsymbol{x}_t\}_{t=0}^{T}$ is determined by the system dynamics (1).

The state-dependent cost is defined as

$$\mathcal{S}(\boldsymbol{V}) = \Phi(\boldsymbol{x}_T) + \sum_{t=0}^{T-1} \ell(\boldsymbol{x}_t), \tag{3}$$

where Φ is the terminal state cost and $l(\boldsymbol{x}_t)$ is the running state cost at time t. Using this cost definition, the SOC problem can be formulated to establish the connection with information theory, as follows:

$$\boldsymbol{U}^* = \arg\min_{\boldsymbol{U}} \ \mathbb{E}_{\mathbb{Q}}\Big[\mathcal{S}(\boldsymbol{V}) + \frac{\lambda}{2}\sum_{t=0}^{T-1}\big(\boldsymbol{u}_t^\top \Sigma^{-1}\boldsymbol{u}_t + \boldsymbol{\beta}_t^\top \boldsymbol{u}_t\big)\Big], \tag{4}$$

where $\boldsymbol{\beta}_t \in \mathbb{R}^m$ is the coefficient vector of the linear control cost term, and $\lambda > 0$ is a weighting parameter that scales the quadratic control cost.

To enable the information-theoretic derivation, we introduce the free-energy functional:

$$\mathcal{F}_\lambda = -\lambda \log \mathbb{E}_{\mathbb{P}}\big[\exp(-\frac{1}{\lambda}\mathcal{S}(\boldsymbol{V}))\big], \tag{5}$$

where $\mathbb{P}$ represents the base distribution over control sequences $\boldsymbol{V}$, typically chosen as zero-mean Gaussian noise: $\boldsymbol{v}_t \sim \mathcal{N}(\boldsymbol{0}, \Sigma)$. Unlike in (4), where λ appeared as a weighting factor for the quadratic control cost, here λ serves as a temperature parameter. By applying Jensen's inequality and using importance sampling, the free-energy in (5) can be upper bounded as follows:

$$\mathcal{F}_\lambda \leq \mathbb{E}_{\mathbb{Q}}\big[\mathcal{S}(\boldsymbol{V})\big] + \lambda \mathbb{D}_{\mathrm{KL}}(\mathbb{Q}\,\|\,\mathbb{P}). \tag{6}$$

To establish the connection with (4), the base distribution is set to $p = q(\boldsymbol{V} \mid \tilde{\boldsymbol{U}}, \Sigma)$ where $\tilde{\boldsymbol{U}}$ represents a reference control sequence, and the KL divergence is expanded as follows:

$$\mathcal{F}_\lambda \leq \mathbb{E}_{\mathbb{Q}}\Big[\Phi(\boldsymbol{x}_T) + \sum_{t=0}^{T-1}\Big(\ell(\boldsymbol{x}_t) + \frac{\lambda}{2}(\boldsymbol{u}_t^\top\Sigma^{-1}\boldsymbol{u}_t + \boldsymbol{\beta}_t^\top\boldsymbol{u}_t + c_t)\Big)\Big] \tag{7}$$

where $\boldsymbol{\beta}_t := -\Sigma^{-1}\tilde{\boldsymbol{u}}_t,\ c_t := \tilde{\boldsymbol{u}}_t^\top\Sigma^{-1}\tilde{\boldsymbol{u}}_t$ represent the linear and constant components resulting from shifting the mean control from $\tilde{\boldsymbol{U}}$ to $\boldsymbol{U}$.

As a result, the cost structure in (7) aligns with that in (4), implying that the free energy serves as a lower bound on the optimal control objective. To attain this bound, the distribution that satisfies equality in (6), (7) is defined as the optimal distribution $\mathbb{Q}^*$:

$$q^*(\boldsymbol{V}) = \frac{1}{\eta}\exp\Big(-\frac{1}{\lambda}\mathcal{S}(\boldsymbol{V})\Big)\,p(\boldsymbol{V}), \tag{8}$$

where

$$\eta = \int \exp\Big(-\frac{1}{\lambda}\mathcal{S}(\boldsymbol{V})\Big)\,p(\boldsymbol{V})\,d\boldsymbol{V} \tag{9}$$

is the normalizing constant. Using this definition, the SOC problem in (4) can be reformulated as minimizing the KL divergence between the optimal distribution $\mathbb{Q}^*$ and the controlled distribution $\mathbb{Q}$:

$$\begin{aligned}
\boldsymbol{U}^* &= \arg\min_{\boldsymbol{U}} \ \mathbb{D}_{\mathrm{KL}}\big(\mathbb{Q}^*\,\|\,\mathbb{Q}\big) \\
&= \arg\min_{\boldsymbol{U}} \ \mathbb{E}_{\mathbb{Q}^*}\Big[\sum_{t=0}^{T-1}(\boldsymbol{v}_t - \boldsymbol{u}_t)^\top\Sigma^{-1}(\boldsymbol{v}_t - \boldsymbol{u}_t)\Big] \qquad (10)\\
&= \mathbb{E}_{\mathbb{Q}^*}\Big[\boldsymbol{V}\Big].
\end{aligned}$$

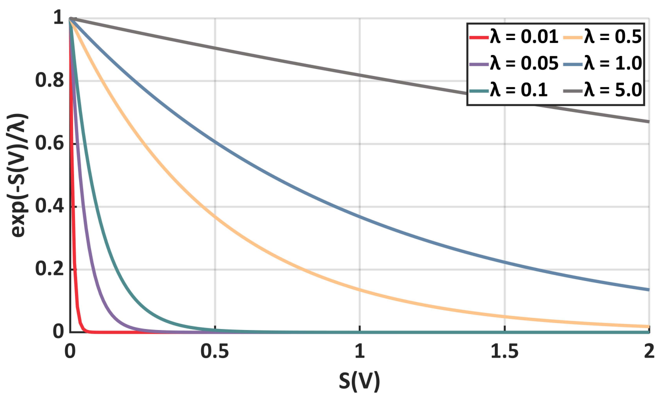


Figure 1: Effect of the temperature parameter λ on the exponential weighting function $\exp(-S/\lambda)$. Smaller λ values yield sharper decay, emphasizing low-cost trajectories, whereas larger λ values produce smoother weighting and higher exploration.

Since direct sampling from the optimal distribution $\mathbb{Q}^*$ is intractable, MPPI employs importance sampling using the Gaussian distribution $q(\boldsymbol{V} \mid \hat{\boldsymbol{U}}, \Sigma)$, where $\hat{\boldsymbol{U}}$ represents the current control estimate. The optimization in (10) can then be reformulated as follows:

$$\begin{aligned}\mathbb{E}_{\mathbb{Q}^*}[\boldsymbol{V}] &= \int q^*(\boldsymbol{V})\, \boldsymbol{V}\, d\boldsymbol{V} \\ &= \int \frac{q^*(\boldsymbol{V})}{q(\boldsymbol{V} \mid \hat{\boldsymbol{U}}, \Sigma)}\, q(\boldsymbol{V} \mid \hat{\boldsymbol{U}}, \Sigma)\, \boldsymbol{V}\, d\boldsymbol{V} \\ &= \mathbb{E}_{\mathbb{Q}}\big[\, w(\boldsymbol{V})\, \boldsymbol{V}\,\big].\end{aligned} \tag{11}$$

The importance weight $w(\boldsymbol{V})$ is derived by factorizing through the base distribution $p(\boldsymbol{V})$:

$$\begin{aligned}w(\boldsymbol{V}) &= \left(\frac{q^*(\boldsymbol{V})}{p(\boldsymbol{V})}\right)\left(\frac{p(\boldsymbol{V})}{q(\boldsymbol{V} \mid \hat{\boldsymbol{U}}, \Sigma)}\right) \\ &= \frac{1}{\eta}\exp\Big(-\tfrac{1}{\lambda}\,\mathcal{S}(\boldsymbol{V})\Big)\, \frac{p(\boldsymbol{V})}{q(\boldsymbol{V} \mid \hat{\boldsymbol{U}}, \Sigma)} \\ &= \frac{1}{\eta}\exp\Big(-\frac{1}{\lambda}\big(\mathcal{S}(\boldsymbol{V}) + \lambda\sum_{t=0}^{T-1}(\hat{\boldsymbol{u}}_t - \tilde{\boldsymbol{u}}_t)^\top \Sigma^{-1}\boldsymbol{v}_t\big)\Big).\end{aligned} \tag{12}$$

The importance weight in (12) is exponentially shaped by the temperature parameter λ, which controls the concentration of the sampling distribution around low-cost trajectories, as illustrated in Fig. 1. Substituting the importance weight derived in (12) into (11) yields the theoretical MPPI control law. Although the solution is theoretically global, it requires evaluating expectations over the entire trajectory space, which is computationally intractable. In practice, this expectation is approximated using Monte Carlo (MC) sampling, resulting in the practical MPPI algorithm that forms the foundation for real-time implementations.

2.2. Classical Path-Integral Derivation of MPPI

This subsection presents the derivation of MPPI from the perspective of classical path integral control theory, establishing its theoretical connection to the HJB framework. The key insight is transforming the nonlinear HJB equation into a linear partial differential equation through exponential transformation, enabling forward sampling of stochastic trajectories rather than backward value function computation. Conceptually, this approach differs from the information-theoretic formulation in Section 2.1 in both sampling domain and optimization output: while the information-theoretic approach samples directly in control space and returns an open-loop control sequence, the path-integral approach samples state trajectories under nominal uncontrolled dynamics and produces single-step control update via cost-weighted averaging. For readers interested in the detailed mathematical foundations of this classical path-integral formulation, we refer to [25].

We begin by formulating a general continuous-time stochastic dynamical system as follows:

$$d\boldsymbol{x}_t = f(\boldsymbol{x}_t)\, dt + G(\boldsymbol{x}_t)\, \boldsymbol{u}_t\, dt + B(\boldsymbol{x}_t)\, d\boldsymbol{w}_t, \tag{13}$$

where $G(\boldsymbol{x}_t) \in \mathbb{R}^{n\times m}$ denotes the state-dependent control transition matrix, $B(\boldsymbol{x}_t) \in \mathbb{R}^{n\times q}$ represents the diffusion matrix, and $d\boldsymbol{w}_t \in \mathbb{R}^q$ is the differential of a standard Brownian motion capturing stochastic disturbances. The state is partitioned into a passive part and a controlled part, and the system components $\boldsymbol{x}, f, G, B$ are expressed as follows:

$$\begin{aligned}\boldsymbol{x}_t &= \begin{bmatrix}\boldsymbol{x}_t^p \\ \boldsymbol{x}_t^c\end{bmatrix}, \qquad & f(\boldsymbol{x}_t) &= \begin{bmatrix}f_p(\boldsymbol{x}_t^p) \\ f_c(\boldsymbol{x}_t^c)\end{bmatrix}, \\ G(\boldsymbol{x}_t) &= \begin{bmatrix}\mathbf{0}_{\ell\times m} \\ G_c(\boldsymbol{x}_t^c)\end{bmatrix}, \qquad & B(\boldsymbol{x}_t) &= \begin{bmatrix}\mathbf{0}_{\ell\times q} \\ B_c(\boldsymbol{x}_t^c)\end{bmatrix}.\end{aligned} \tag{14}$$

Only the controlled part $\boldsymbol{x}_t^c$ is directly influenced by control inputs and process noise, whereas the passive part $\boldsymbol{x}_t^p$ evolves deterministically without direct actuation.

To derive the path integral formulation, let us first establish the classical stochastic optimal control framework. The value function, which represents the optimal expected future cost from state $\boldsymbol{x}_t$ at time t, is defined as

$$V(\boldsymbol{x}_t, t) = \min_{\boldsymbol{U}_{t:T}} \mathbb{E}_{\mathbb{Q}}\left[\phi(\boldsymbol{x}_T) + \int_t^T \Big(\ell(\boldsymbol{x}_t) + \tfrac{1}{2}\,\boldsymbol{u}_t^\top R\,\boldsymbol{u}_t\Big)\, dt\right], \tag{15}$$

where $R \in \mathbb{R}^{m\times m}$ is positive definite. Given the stochastic optimal control problem defined by (13)–(15), the optimal value function $V(\boldsymbol{x}_t, t)$ satisfies the stochastic HJB equation,

$$\begin{aligned}-\partial_t V(\boldsymbol{x}_t, t) &= \ell(\boldsymbol{x}_t) + f(\boldsymbol{x}_t)^\top \nabla_{\boldsymbol{x}} V(\boldsymbol{x}_t, t) \\ &\quad - \frac{1}{2}\nabla_{\boldsymbol{x}} V(\boldsymbol{x}_t, t)^\top G(\boldsymbol{x}_t) R^{-1} G(\boldsymbol{x}_t)^\top \nabla_{\boldsymbol{x}} V(\boldsymbol{x}_t, t) \\ &\quad + \frac{1}{2}\mathrm{tr}\big(B(\boldsymbol{x}_t) B(\boldsymbol{x}_t)^\top \nabla^2_{\boldsymbol{x}\boldsymbol{x}} V(\boldsymbol{x}_t, t)\big),\end{aligned} \tag{16}$$

with the boundary condition,

$$V(\boldsymbol{x}_T, T) = \phi(\boldsymbol{x}_T). \tag{17}$$

The optimal control law can be obtained by solving the HJB equation as follows:

$$\boldsymbol{u}^*(\boldsymbol{x}_t, t) = -R^{-1} G(\boldsymbol{x}_t)^\top \nabla_{\boldsymbol{x}} V(\boldsymbol{x}_t, t). \tag{18}$$

Although the HJB equation (16) provides a principled way to compute the optimal control law (18), it must be solved backward in time from the terminal state to the initial state. This backward propagation requires evaluating the value function over the entire state-time space, which quickly becomes computationally intractable as the state dimension increases, commonly referred to as the curse of dimensionality. To overcome this issue, the nonlinear HJB equation can be transformed into a linear partial differential equation, allowing the use of forward sampling of stochastic trajectories. This transformation can be achieved by applying an exponential logarithmic mapping to the value function. Specifically, the desirability function $\Psi(\boldsymbol{x}_t, t)$ is defined as:

$$V(\boldsymbol{x}_t, t) = -\lambda \log \Psi(\boldsymbol{x}_t, t). \tag{19}$$

By differentiating (19) with respect to t and x_t and by computing the Hessian, the following expressions are obtained as follows:

$$\begin{aligned} \partial_t V &= -\frac{\lambda}{\Psi} \frac{\partial \Psi}{\partial t}, \quad \nabla_{\boldsymbol{x}} V = -\frac{\lambda}{\Psi} \nabla_{\boldsymbol{x}} \Psi, \\ \nabla^2_{\boldsymbol{x}\boldsymbol{x}} V &= -\frac{\lambda}{\Psi} \nabla^2_{\boldsymbol{x}\boldsymbol{x}} \Psi + \frac{\lambda}{\Psi^2} \nabla_{\boldsymbol{x}} \Psi \, \nabla_{\boldsymbol{x}} \Psi^\top. \end{aligned} \tag{20}$$

Substituting (20) into the HJB equation (16), the equation can be rewritten as a PDE with respect to the desirability function $\Psi(\boldsymbol{x}_t, t)$ as follows:

$$\begin{aligned} \partial_t \Psi(\boldsymbol{x}_t, t) &= \frac{\Psi(\boldsymbol{x}_t, t)}{\lambda} \ell(\boldsymbol{x}_t) - f(\boldsymbol{x}_t)^\top \nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x}_t, t) \\ &- \frac{\lambda}{2\,\Psi(\boldsymbol{x}_t, t)} \nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x}_t, t)^\top G(\boldsymbol{x}_t) R^{-1} G(\boldsymbol{x}_t)^\top \nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x}_t, t) \\ &+ \frac{1}{2\Psi(\boldsymbol{x}_t, t)} \nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x}_t, t)^\top B(\boldsymbol{x}_t) B(\boldsymbol{x}_t)^\top \nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x}_t, t) \\ &- \frac{1}{2} \operatorname{tr}\big(B(\boldsymbol{x}_t) B(\boldsymbol{x}_t)^\top \nabla^2_{\boldsymbol{x}\boldsymbol{x}} \Psi(\boldsymbol{x}_t, t)\big). \end{aligned} \tag{21}$$

Under the noise–control matching assumption

$$B(\boldsymbol{x}_t) B(\boldsymbol{x}_t)^\top = \lambda\, G(\boldsymbol{x}_t)\, R^{-1}\, G(\boldsymbol{x}_t)^\top, \tag{22}$$

the quadratic gradient terms in (21) cancel, and the HJB in terms of the desirability becomes a linear PDE. The process noise is assumed to act along the same control-effective channels as $G(\boldsymbol{x}_t)$, and its covariance is scaled consistently with the control cost R. As a result, the desirability function $\Psi(\boldsymbol{x}_t, t)$ satisfies the following linear partial differential equation:

$$\begin{aligned} \partial_t \Psi(\boldsymbol{x}_t, t) &= \frac{\Psi(\boldsymbol{x}_t, t)}{\lambda} \ell(\boldsymbol{x}_t) - f(\boldsymbol{x}_t)^\top \nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x}_t, t) \\ &- \frac{1}{2} \operatorname{tr}\Big(B(\boldsymbol{x}_t) B(\boldsymbol{x}_t)^\top \nabla^2_{\boldsymbol{x}\boldsymbol{x}} \Psi(\boldsymbol{x}_t, t)\Big), \end{aligned} \tag{23}$$

where the terminal condition is given by

$$\Psi(\boldsymbol{x}_T, T) = \exp\big(-\phi(\boldsymbol{x}_T)/\lambda\big). \tag{24}$$

Applying the Feynman–Kac theorem to the linear PDE in (23) yields the solution as follows:

$$\begin{aligned} \Psi(\boldsymbol{x}_{t_0}, t_0) &= \mathbb{E}_{\mathbb{P}}\left[\exp\left(-\frac{1}{\lambda}\int_{t_0}^{T} \ell(\boldsymbol{x}_t)\, dt\right) \Psi(\boldsymbol{x}_T, T)\right] \\ &= \mathbb{E}_{\mathbb{P}}\left[\exp\left(-\frac{1}{\lambda} S(\tau)\right)\right], \end{aligned} \tag{25}$$

where $S(\tau) = \phi(\boldsymbol{x}_T) + \int_{t_0}^{T} \ell(\boldsymbol{x}_t)\, dt$ denotes the total state cost along the trajectory. By differentiating (25) and using (19), then substituting the resulting expression into the optimality condition (18), the optimal control law can be obtained in continuous time as follows:

$$\boldsymbol{u}^*(\boldsymbol{x}_t, t)\, dt = \mathcal{G}(\boldsymbol{x}_t) \frac{\mathbb{E}_{\mathbb{P}}\big[\exp\big(-\frac{1}{\lambda} S(\tau)\big)\ d\boldsymbol{w}\big]}{\mathbb{E}_{\mathbb{P}}\big[\exp\big(-\frac{1}{\lambda} S(\tau)\big)\big]}, \tag{26}$$

where

$$\mathcal{G}(\boldsymbol{x}_t) = R^{-1} G_c(\boldsymbol{x}_t)^\top \big(G_c(\boldsymbol{x}_t) R^{-1} G_c(\boldsymbol{x}_t)^\top\big)^{-1} B_c(\boldsymbol{x}_t). \tag{27}$$

The continuous time control law in (26) can be further written in discrete time using the Euler–Maruyama scheme, with $d\boldsymbol{w} = \boldsymbol{\epsilon}\sqrt{\Delta t}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, as follows:

$$\boldsymbol{u}^*(\boldsymbol{x}_t, t) = \mathcal{G}(\boldsymbol{x}_t) \frac{\mathbb{E}_{\mathbb{P}}\Big[\exp\big(-\frac{1}{\lambda} S(\tau)\big)\ \frac{\boldsymbol{\epsilon}}{\sqrt{\Delta t}}\Big]}{\mathbb{E}_{\mathbb{P}}\big[\exp\big(-\frac{1}{\lambda} S(\tau)\big)\big]}. \tag{28}$$

By setting the time step to $\Delta t = 1$ without loss of generality, and making the additional assumption that the noise enters the system through the control input as $B = G\sqrt{\Sigma}$ with $R = \lambda \Sigma^{-1}$, the control law in (28) can be further simplified as follows:

$$\boldsymbol{u}^*(\boldsymbol{x}_t, t) = \frac{\mathbb{E}_{\mathbb{P}}\Big[\exp\big(-\frac{1}{\lambda} S(\tau)\big)\ \sqrt{\Sigma}\,\boldsymbol{\epsilon}\Big]}{\mathbb{E}_{\mathbb{P}}\big[\exp\big(-\frac{1}{\lambda} S(\tau)\big)\big]}. \tag{29}$$

In summary, the exponential transformation (19) combined with the noise-control matching assumption (22) successfully linearizes the nonlinear HJB equation, yielding the path integral control law (29). A key technical insight is that the quadratic control penalty disappears from the final expression, as control costs are implicitly encoded through the noise covariance structure via the matching condition $BB^\top = \lambda G R^{-1} G^\top$. This mechanism naturally adapts the control effort based on the noise characteristics: lower costs in high-noise directions and higher costs in low-noise directions. The resulting formulation provides an alternative theoretical foundation for MPPI that complements the information-theoretic perspective presented in Section 2.1, demonstrating the rich mathematical structure underlying this sampling-based control method.

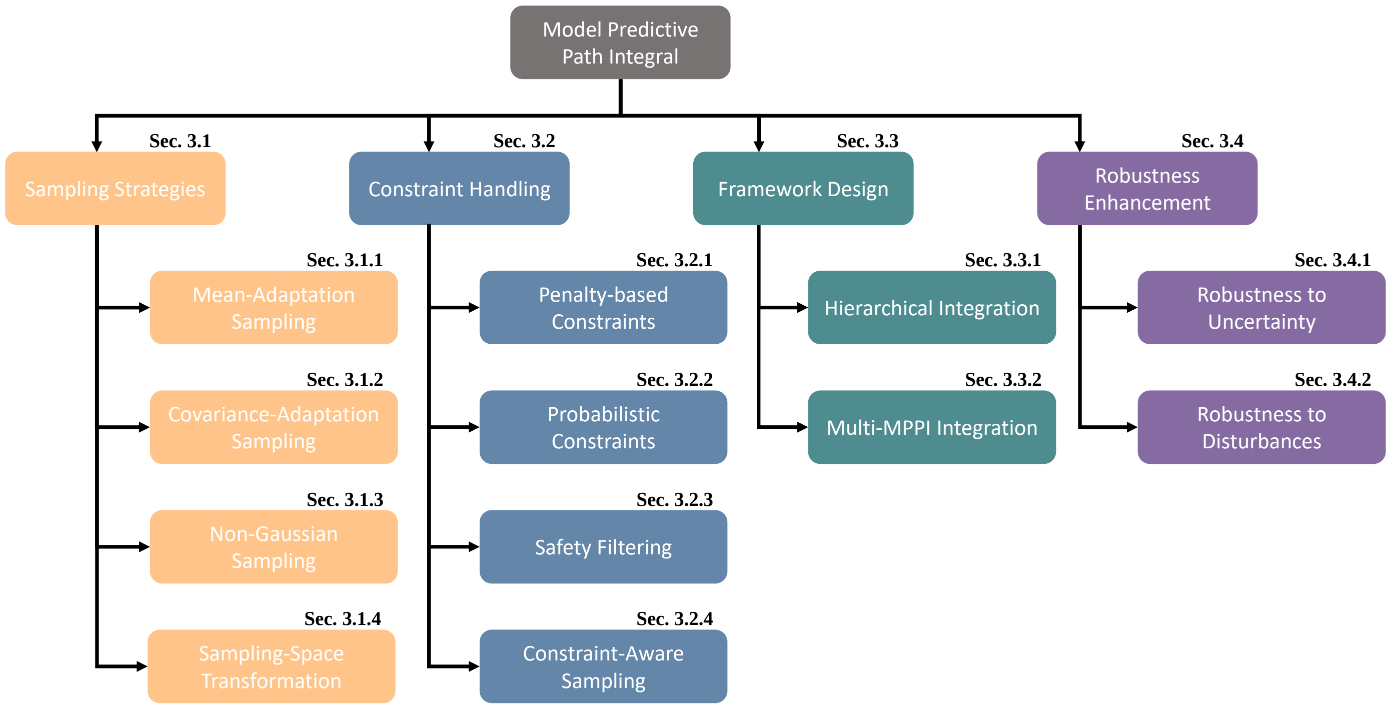


Figure 2: Taxonomy of methodological themes in MPPI.

3. Methodological Themes of MPPI

This section provides a structured review of methodological advances in MPPI research. To bring clarity to the fragmented landscape of MPPI variants, we categorize existing approaches into four key themes: **(i) Sampling Strategies, (ii) Constraint Handling, (iii) Framework Design, and (iv) Robustness Enhancement**. As illustrated in Fig. 2, these four overarching themes encompass the principal methodological directions along which approaches built upon MPPI have been developed and diversified. Within each theme, existing works can be further subdivided into more specific categories that capture distinct algorithmic trends. For example, Sampling Strategies in this survey are categorized into mean-adaptation sampling, covariance-adaptation sampling, non-Gaussian sampling, and sampling-space transformation. The following subsections elaborate on each theme and its representative methods in detail.

3.1. Sampling Strategies

One of the core factors determining MPPI performance is the quality of sampled trajectories. Since MPPI relies on importance-weighted averaging over sampled rollouts, its efficiency depends on whether low-cost trajectories are sufficiently explored within a limited sample budget. This reliance is explicitly reflected in the standard MPPI update law, where the nominal control sequence is refined by aggregating perturbations weighted by their exponentiated utility:

$$\boldsymbol{u}_t^* = \boldsymbol{u}_t + \sum_{k=1}^{K} w^{(k)}\, \delta\boldsymbol{u}_t^{(k)}, \tag{30}$$

where $\boldsymbol{u}_t$ represents the nominal mean control, $\delta\boldsymbol{u}_t^{(k)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ denotes the sampled control perturbation, $w^{(k)}$ is the corresponding importance weight defined in (12), and K is the number of sampled trajectories. However, purely random Gaussian sampling often fails to generate sufficient low-cost trajectories, resulting in limited coverage of promising regions and poor scalability for high-dimensional or long-horizon tasks. To address these limitations, numerous studies have sought to redesign the sampling distribution for more effective exploration.

3.1.1. Mean-Adaptation Sampling

A major limitation of the standard MPPI control framework lies in the use of a naive Gaussian sampling distribution centered around the control sequence obtained in the previous iteration. Such uninformed sampling typically results in inefficient trajectory exploration, particularly when applied to complex environments or high-dimensional nonlinear dynamical systems. To mitigate this limitation, a class of mean-adaptation sampling methods has been developed to directly modify the nominal control sequence u_t in the MPPI update (30), thereby biasing the proposal distribution toward dynamically feasible and cost-efficient regions. By explicitly defining or adaptively updating a reference mean control sequence, these methods shift the center of the sampling distribution to guide exploration toward more promising trajectories.

A straightforward way to implement this approach is to inject an external path planner that provides a nominal control sequence serving as the mean of the sampling distribution. For example, Tao *et al.* [26] proposed a Rapidly-exploring Random Tree (RRT)-guided MPPI algorithm, in which a nominal trajectory computed by the RRT is converted into a corresponding control sequence and subsequently used as the mean of the Gaussian proposal distribution. Likewise, Yang *et al.* [27] extended the same concept through a traversability-aware Hybrid A* planner operating on a 2.5-dimensional cost map, which generates a nominal control sequence that improves vehicle stability and reduces pitch and roll oscillations on uneven terrains. Although the integration of deterministic motion planners provides initial guidance and accelerates early convergence, the planner-guided MPPI approach inherently requires an explicit environmental model.

This model dependence led to the development of data-driven mean generation techniques, where the sampling mean is inferred from data rather than constructed analytically. Huang *et al.* [28] introduced Diffusion-MPPI, which employs a conditional diffusion model to generate a nominal trajectory that initializes the mean of the sampling distribution from environmental point-cloud observations. The diffusion model provides robust distributional guidance but requires iterative denoising, which increases computational cost during online control. In contrast, Kusumoto et al. [29] proposed Informed Information-Theoretic MPC (IIT-MPC), which adopts a single-shot latent decoding approach based on a Conditional Variational Autoencoder (CVAE) to generate a context-aware mean trajectory with significantly reduced online computational overhead. Alternatively, Qu *et al.* [30] presented RL-driven MPPI, where an offline-trained reinforcement learning policy is used to initialize the sampling mean. Despite being computationally lightweight, this policy-based guidance may introduce bias toward a single behavioral mode and suffer from limited adaptability under distributional shift. To address this limitation, Wang *et al.* [31] proposed Residual-MPPI, which not only uses a pretrained policy as the mean initializer but also incorporates the policy's log-likelihood into the MPPI objective. This formulation enables principled online policy customization, allowing the controller to preserve desirable prior behaviors while adapting to new task objectives or distributional changes.

In contrast to the above data-driven approaches that preserve sampling diversity through generative priors or learned policies, deterministic mode-seeking methods have been explored to directly concentrate samples around dominant low-cost solutions. Honda *et al.* [32] proposed Stein Variational Guided MPPI (SVG-MPPI), which iteratively transports samples toward low-cost regions using Stein Variational Gradient Descent (SVGD). Unlike diffusion- or CVAE-based methods that preserve sample diversity, SVG-MPPI performs deterministic mode-seeking transport, progressively aligning the sampling mean with a dominant feasible mode. This improves convergence stability in scenarios with sharply separated solution modes.

3.1.2. Covariance-Adaptation Sampling

Regulating the covariance structure of the sampling distribution is a key strategy that goes beyond merely reducing noise, aiming to improve the quality of exploration and convergence stability. These approaches can be broadly categorized into methods that design the covariance structure to introduce temporal correlation, and those that dynamically regulate the magnitude of variance to balance exploration and exploitation.

One class of approaches focuses on introducing temporal coherence to the sampled rollouts. Time-Correlated MPPI (TC-MPPI), introduced by Lee *et al.* [33], incorporates a quadratic cost on action derivatives into the optimal control problem. This cost term is mathematically absorbed into the covariance matrix, creating a non-diagonal inverse covariance matrix that encodes correlations between temporally adjacent control inputs. As a result, the sampling process itself embeds temporal correlation, leading to dynamically feasible and smoother trajectories. Low-Frequency MPPI, proposed by Vlahov *et al.* [34], achieves this goal differently by sampling colored noise in the frequency domain. In this approach, the variance of each frequency component is scaled according to a power law to intentionally suppress high-frequency components. The resulting control samples, generated via an inverse Fast Fourier Transform (iFFT), are inherently smooth and dominated by low frequencies, providing trajectories that are better suited for long-horizon exploration.

Another class of approaches regulates the magnitude of the variance based on real-time feedback or a predefined schedule. Dynamic-Variance MPPI (DV-MPPI), presented by Kim *et al.* [35], employs an intuitive mechanism that scales the variance in proportion to the real-time tracking error using a hyperbolic tangent function. This mechanism allows for aggressive exploration when the error is large and stable, fine-grained control when the error is small. However, because variance adaptation is triggered only after tracking error is observed, DV-MPPI reacts to disturbances rather than anticipating them, which may temporarily limit exploration in rapidly changing environments. As an alternative variance modulation strategy, Xue *et al.* [36] proposed Diffusion-Inspired Annealing for Legged MPC (DIAL-MPC), which avoids reliance on feedback by employing a proactive annealing schedule. Rather than waiting for error feedback, DIAL-MPC progressively decreases the variance through a diffusion-inspired dual-loop process within a single control step, maintaining exploratory capability while guiding rollouts toward local convergence.

Beyond these two covariance-level adaptation strategies, a distinct line of work seeks to optimize the entire sampling distribution directly. Model Predictive Optimized Path Integral (MPOPI), proposed by Asmar et al. [37], formulates the control sequence as a single joint vector and ap-

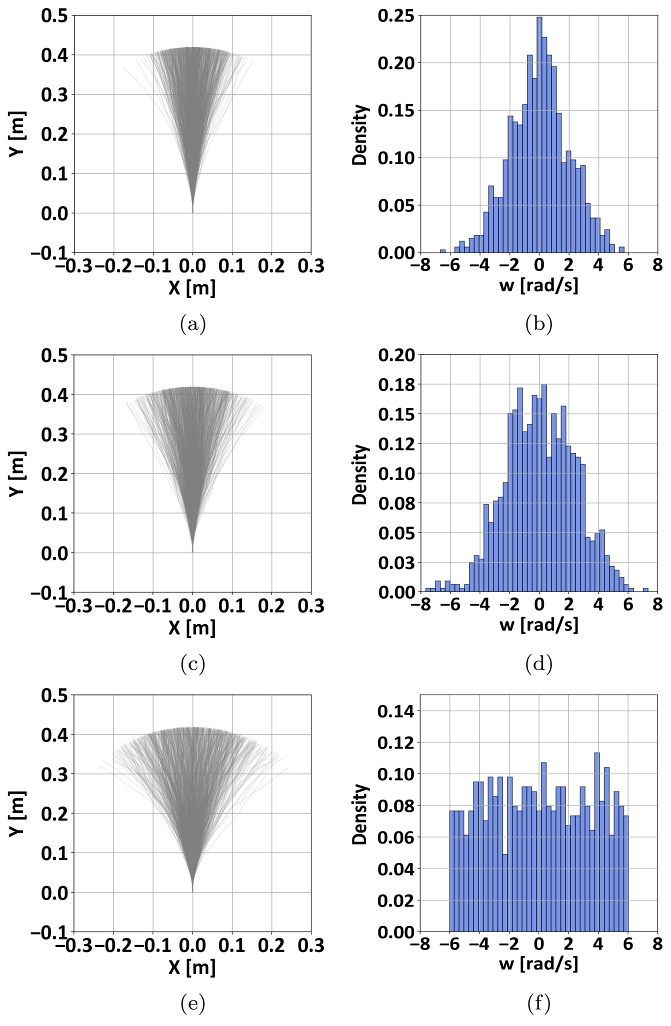


Figure 3: Visualization of 1,000 sampled control noises and the corresponding trajectories under different sampling distributions. (a) Histogram of control-input noise sampled from a Gaussian distribution based on [24]. (b) Resulting trajectories obtained from the Gaussian sampling, where the control noise follows $\delta\boldsymbol{u} \sim \mathcal{N}(0, 2.0^2)$. (c) Histogram of control-input noise sampled from an NLN distribution based on [38]. (d) Resulting trajectories obtained from the NLN sampling, which consists of a normal component $\delta\boldsymbol{u}^n \sim \mathcal{N}(0, 2.0^2)$ and a lognormal component $\delta\boldsymbol{u}^{nl} \sim \mathrm{Log}\mathcal{N}(0.2, 0.1^2)$. (e) Histogram of control-input noise sampled from a random uniform distribution inspired by [39]. (f) Resulting trajectories obtained from the uniform sampling with a range of $[-6, 6]$.

plies Adaptive Importance Sampling (AIS). This process iteratively updates both the mean and covariance of the joint distribution within a single control step, significantly improving sample efficiency by more effectively targeting high-reward regions.

3.1.3. Non-Gaussian Sampling

Section 2 provides an information-theoretic interpretation of the stochastic optimal control problem and introduces the free-energy inequality shown in (6). When both $\mathbb{P}$ and $\mathbb{Q}$ are assumed to be Gaussian, the problem becomes equivalent to the SOC formulation; however, this assumption is not essential. From an information-theoretic standpoint, the optimal distribution $\mathbb{Q}^*$ that minimizes the free-energy bound remains invariant regardless of the parametric form of $\mathbb{P}$ and $\mathbb{Q}$. Even when $\mathbb{P}$ and $\mathbb{Q}$ are not Gaussian, the KL minimization problem that seeks to align the controlled distribution $\mathbb{Q}$ with the optimal distribution $\mathbb{Q}^*$ remains equivalent under the same information-theoretic formulation. By extending the sampling distribution $\mathbb{Q}$ beyond the Gaussian family to a non-Gaussian form, the search space of the controller is generalized to a broader probabilistic control space, thereby enhancing its capability to approximate optimality across diverse distributional structures.

Early work focused on replacing Gaussian noise $\delta\boldsymbol{u}_t$ in the MPPI update (30) with analytically designed non-Gaussian perturbations to increase sampling diversity. Mohamed *et al.* [38] proposed Log-MPPI, in which the control perturbation $\delta\boldsymbol{u}_t$ is constructed from a Normal–LogNormal (NLN) mixture distribution to introduce skewness and heavy tails:

$$\delta\boldsymbol{u}_t = \delta\boldsymbol{u}_t^n \cdot \delta\boldsymbol{u}_t^{ln}, \tag{31}$$

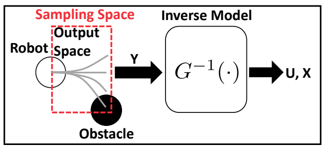
where

$$\delta\boldsymbol{u}_t^n \sim \mathcal{N}\big(\mathbf{0}, \boldsymbol{\Sigma}^n\big), \qquad \delta\boldsymbol{u}_t^{ln} \sim \mathrm{Log}\mathcal{N}\big(\boldsymbol{\mu}^{ln}, \boldsymbol{\Sigma}^{ln}\big). \tag{32}$$

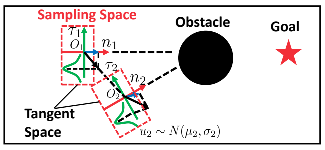
This mixture-based sampling expands the exploration range beyond Gaussian perturbations and improves robustness against local minima. Extending this direction, Seo *et al.* [39] replace Gaussian noise with a uniform distribution-based deterministic sampling scheme, significantly reducing the number of samples required for real-time CoM trajectory generation.

Fig. 3 illustrates the control-input noise distributions and the resulting state trajectories for Gaussian, NLN, and Uniform sampling schemes using comparable scales. Distinct from the deterministic scheme employed in [39], we adopt random uniform sampling to ensure a fair evaluation under stochastic variability. In terms of trajectory generation, the NLN distribution yields a wider spread due to its skewness, despite sharing the same mean and variance as the Gaussian case. In contrast, the Uniform sampling provides perturbations with uniform density, leading to improved exploration efficiency but still resulting in limited state-space coverage after dynamics propagation. To address this state-space coverage limitation and the tendency of samples to collapse around the nominal action sequence, Poyrazoglu *et al.* [40] proposed C-Uniform trajectory sampling, which explicitly enforces uniform coverage in the configuration space by constructing trajectories that are uniformly distributed over reachable sets. This approach mitigates sample collapse and improves global exploration in motion planning tasks. Nevertheless, these non-Gaussian and uniform-based sampling schemes still do not incorporate explicit task-aware or safety-aware directional guidance in the state space.

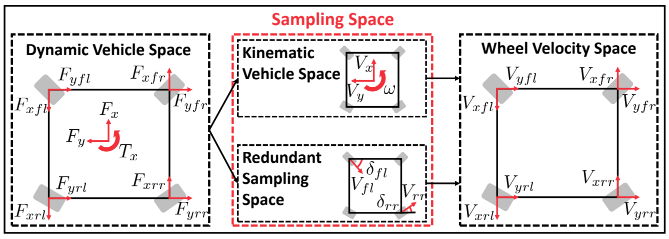
To further improve sampling efficiency, another line of research introduces task-aware bias into the proposal distribution. Homburger *et al.* [41] presented Feature-Based



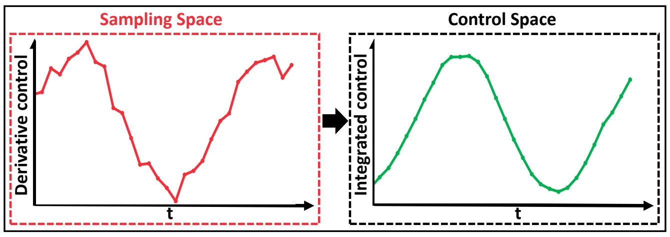
(a)



(b)



(c)



(d)

Figure 4: Overview of MPPI approaches that modify or transform the sampling space to improve performance under different objectives. (a) o-MPPI [43] samples directly in the output trajectory space to increase the likelihood of generating samples that satisfy output constraints. (b) MPPI-Tan [44] restricts sampling onto the obstacle-tangent manifold, suppressing normal-direction perturbations that typically lead to collisions in cluttered scenes. (c) MPPI-H [45] switches between low-dimensional kinematic and redundant wheel-space sampling representations to balance sample efficiency and navigation stability for high-DoF swerve-drive robots. (d) SMPPI [46] lifts the sampling space to derivative-action coordinates, producing smooth control sequences without external filtering in rapidly changing environments.

MPPI, which constructs a mixture proposal by combining task-specific feature policies. This approach injects structured behavioral guidance, such as target tracking or obstacle avoidance, into sampling, broadening exploration in meaningful directions. Similarly, Trevisan and Alonso-Mora [42] proposed Biased-MPPI, which generalizes MPPI to arbitrary proposal distributions by integrating multiple ancillary controllers (e.g., goal-seeking, braking, or safety controllers). These behavior priors are fused through importance-weighted sampling, yielding greater robustness under local minima and dynamic disturbances. These mixture-based methods, however, require manually designed guidance policies, limiting adaptability to new environments. More recent approaches aim to adapt the sampling distribution automatically using data-driven refinement. Flow-based methods such as those by Sacks and Boots [47] and Power and Berenson [48] employ normalizing flows to learn expressive multimodal sampling distributions conditioned on the environment and task context. While the approach in [47] performs online adaptation by updating a latent sampling distribution through bi-level optimization, the method in [48] amortizes a conditional control posterior and additionally introduces an environment-projection mechanism to address out-of-distribution scenarios.

3.1.4. Sampling-Space Transformation

In contrast to approaches that modify the sampling distribution within the original control-input space, a fundamentally different strategy in MPPI is to transform the sampling space itself. By redefining the representation in which control perturbations are generated, these methods enable exploration to be conducted in a coordinate system that more directly reflects task structure, system geometry, or actuation dynamics.

Yan and Devasia [43] proposed Output-Sampled MPPI (o-MPPI), which replaces input-space sampling with sampling directly in the output space, where task-relevant motions such as lane following or path tracking are more naturally represented. By generating rollouts in this transformed domain and mapping them back through an inverse model, o-MPPI improves the compatibility between sampled rollouts and high-level task objectives, without explicitly enforcing constraints. For motion planning around nonconvex obstacles, Zhao *et al.* [44] developed MPPI-Tan, which reparameterizes the control velocity as a weighted combination of nominal and tangential components defined on the obstacle tangent space. By sampling control perturbations in this geometry-aligned coordinate system, MPPI-Tan biases exploration toward locally feasible directions that remain tangent to obstacle boundaries. This geometric reparameterization significantly improves sampling efficiency and robustness in cluttered environments, particularly by mitigating stagnation near local minima.

To address sampling inefficiency in high-dimensional systems, Aoki *et al.* [45] proposed a switching sampling-space strategy for MPPI that alternates between a reduced kinematic space and a slightly redundant control space. By adaptively selecting the sampling representation according to the maneuvering condition, this approach improves stability and navigation performance while maintaining efficient exploration in high-dimensional vehicle control. Furthermore, transforming the sampling domain into the derivative space offers an effective solution to the chattering problem inherent in stochastic sampling. Kim *et al.* [46] proposed Smooth MPPI (SMPPI), which decouples the sampling space from the actual actuation space via an input-lifting strategy. Instead of directly sampling the control inputs, this approach lifts the sampling domain to the derivative action space and recovers the actual control command through temporal integration. By generating samples in this domain, high-frequency noise is inherently smoothed out during the integration process, ensuring smooth control authority without relying on external post-processing filters. These sampling-space transformation strategies are conceptually summarized in Fig. 4.

Table 1: Qualitative comparison of constraint-handling strategies in MPPI.

Criterion	**Penalty-based Constraints** (Section 3.2.1)	**Probabilistic Constraints** (Section 3.2.2)	**Safety Filtering** (Section 3.2.3)	**Constraint-Aware Sampling** (Section 3.2.4)
Sample efficiency	△	△	△	○
Safety guarantees	△	○	○	○
Handling non-differentiable costs	○	△	×	×
Handling of uncertainty	×	○	△	×

○: High, △: Medium, ×: Low.

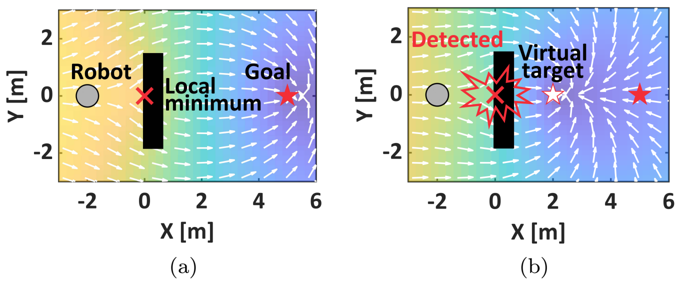


Figure 5: Comparison of guidance fields in repulsive-potential–based MPPI frameworks. (a) RPA-MPPI [49], where a fixed global repulsive potential is applied to push the robot away from the local minimum. (b) DRPA-MPPI [50], which activates the repulsive potential only when a local-minimum condition is detected and redirects the robot toward a dynamically generated virtual target.

3.2. Constraint Handling

MPPI offers the flexibility to encode diverse objectives such as goal tracking, control smoothness, actuation limits, workspace boundaries, and collision avoidance directly through cost terms without requiring model linearization or differentiability. However, since these requirements are typically imposed as soft penalties rather than explicit constraints, MPPI does not provide a formal guarantee of constraint satisfaction, and safety or feasibility violations may still occur. To address this limitation, a wide spectrum of constraint-handling strategies has been developed, ranging from penalty-based formulations to constraint-aware sampling. The following sections review these approaches in detail, while Table 1 presents a structured qualitative comparison of their key characteristics.

3.2.1. Penalty-based Constraints

The most intuitive strategy for handling constraints in MPPI is to embed them directly into the cost function as penalty terms. While this soft-constraint approach does not guarantee strict feasibility, it biases the stochastic optimization process toward safer trajectories by assigning higher costs to constraint-violating regions. Penalty-based formulations are particularly attractive in MPPI for several reasons. They impose no differentiability requirements, readily accommodate black-box or neural-network-based costs, and allow heterogeneous and highly expressive constraint objectives to be incorporated in a unified manner. This general strategy has been explored across a diverse range of constraint types, from spatial constraints to more abstract formulations that shape long-horizon behavior.

Among these applications, the penalty-based approach has been extensively applied to manage spatial constraints, particularly for collision avoidance. In this context, Testouri *et al.* [51] proposed a straightforward yet effective formulation by approximating obstacles as circles with safety margins and directly embedding them into the MPPI cost function. While such direct penalty formulations are effective for basic collision avoidance, they may inadvertently introduce local minima in environments with complex or nonconvex obstacles. To explicitly address entrapment in such local minima, Fuke *et al.* [49] introduced Repulsive Potential Augmented-MPPI (RPA-MPPI), which adds an artificial repulsive potential field to the cost function as a static penalty term to push trajectories away from trap regions. However, because the repulsive potential is always active regardless of the actual entrapment risk, RPA-MPPI can lead to overly conservative or inefficient detours in benign situations. To overcome this limitation, they later proposed Dynamic Repulsive Potential Augmented-MPPI (DRPA-MPPI) [50], which dynamically detects imminent local-minimum entrapment and activates a detour-inducing repulsive cost only when necessary. The behavioral difference between RPA-MPPI and DRPA-MPPI is illustrated in Fig. 5, where the former continuously applies a static repulsive influence, while the latter generates a virtual detour target only upon detecting a local-minimum condition. Broadening the scope from binary collision penalties and local-minimum avoidance to continuous spatial risk modeling, Ardiyanto *et al.* [52] proposed Eikonal-MPPI, which integrates a risk transform derived from the eikonal equation into the MPPI cost structure. This approach provides a compact representation of spatial risk and guides trajectories away from hazardous regions in a smooth and globally informed manner.

Beyond spatial constraints, penalty-based formulations have also been extended to more general constraint structures and long-horizon reasoning. In this context, Crestaz *et al.* [53] proposed Temporal-Difference Constraint-Discounted-MPPI (TD-CD-MPPI), which handles constraints by modulating the discount factor of sampled trajectories according to their degree of constraint violation and augmenting the return with a terminal value function learned via temporal-difference updates. Through this constraint-discounted return formulation, trajectories that incur constraint violations are continuously down-weighted over the prediction horizon, enabling soft constraint handling without explicit barrier functions, projections, or feasibility filtering.

3.2.2. Probabilistic Constraints

Beyond penalty-based formulations, probabilistic constraint handling aims to regulate safety under uncertainty by explicitly considering stochastic variability in the system and environment. Unlike deterministic constraints, these approaches reason about risk at the distributional level and seek to control either the likelihood or the severity of constraint violations. A variety of such formulations have been explored in the MPPI literature, ranging from implicit distribution shaping to explicit probabilistic constraints and risk-sensitive objectives.

An early line of work regulates probabilistic risk implicitly by shaping the state distribution, without introducing explicit chance constraints. Yin *et al.* [54] proposed the Covariance-Controlled MPPI (CC-MPPI), which augments the standard MPPI formulation with an explicit terminal covariance bound. By propagating and shaping the state covariance through the dynamics using covariance steering, CC-MPPI indirectly limits the dispersion of sampled trajectories and improves robustness under stochastic disturbances, while preserving the information-theoretic structure of MPPI. Although no explicit probability constraint is imposed, the reduction of terminal uncertainty effectively suppresses the probability of large stochastic deviations.

The above implicit regulation naturally motivates the explicit formulation of probabilistic safety through chance constraints. A common formalization enforces that a state-dependent constraint function $g(x)$ is satisfied with high probability:

$$\Pr\left[g(\boldsymbol{x}_k) \leq 0,\ \forall k \in [0, N]\right] \geq 1 - \delta, \tag{33}$$

where δ specifies the admissible level of risk. This formulation enables a direct trade-off between safety and performance but also introduces significant computational challenges. Several recent works further consider scenarios in which model uncertainty must be learned directly from data. Trivedi *et al.* [55] proposed a Gaussian-Process-based chance-constrained MPPI framework, in which safety is conceptually formulated as a chance constraint but enforced deterministically through GP-derived covariance-based margin tightening. By adapting safety margins to model uncertainty and leveraging GPU-parallelized sampling, this approach achieves real-time, risk-aware navigation in highly uncertain environments. Although the chance constraint appears in the problem formulation, the actual implementation avoids explicit probability integration by relying on deterministic uncertainty bounds.

In contrast, Balci *et al.* [56] introduced the Constrained Covariance Steering based Tube-MPPI (CCS-MPPI) as an explicit realization of chance constraints under linear Gaussian assumptions. By reformulating chance constraints into deterministic convex conditions and constructing a probabilistically safe tube around a nominal trajectory, CCS-MPPI provides formal probabilistic safety guarantees. However, this formulation is restricted to stochastic linear systems, which limits its applicability to general nonlinear dynamics. To overcome this limitation and extend chance-constrained MPPI to nonlinear systems, Yin *et al.* [57] proposed the Belief-Space Stochastic MPPI (BSS-MPPI). This method preserves the full nonlinear system dynamics by leveraging Monte Carlo (MC) forward simulation for state propagation. However, to make the probabilistic safety check computationally tractable, it approximates the chance constraint boundary via local linearization with a conservative back-off in the belief space. Similarly, Mohamed *et al.* [58] introduced the Chance-Constrained Unscented MPPI (C2U-MPPI), which replaces gradient-based linearization with the Unscented Transform to propagate uncertainty through nonlinear dynamics and evaluates collision risk using a Mahalanobis-distance-based criterion with a dynamic threshold. These methods enable deterministic approximations of chance constraints for nonlinear systems.

Instead of relying on analytic or deterministic approximations, an alternative direction directly estimates collision probability via sampling. Trevisan *et al.* [59] proposed the Dynamic Risk-Aware MPPI (DRA-MPPI), which approximates the joint collision probability for each rollout using efficient parallel MC sampling. This risk estimate is used either as a cost penalty or to reject high-risk samples from the MPPI update. By avoiding Gaussian assumptions and linearization, DRA-MPPI can handle arbitrary, non-Gaussian obstacle predictions, at the expense of increased computational overhead, which is mitigated through massive parallelization.

Finally, rather than bounding violation probability itself, a complementary class of methods focuses on quantifying the severity of rare events using coherent risk measures. A representative example is the Conditional Value-at-Risk (CVaR), which measures the expected cost of the worst $(1-\alpha)$ outcomes, where the tail probability $1-\alpha$ plays a role analogous to the admissible risk level δ in (33). Generally defined for a random cost variable Z, it is expressed as:

$$\mathrm{CVaR}_\alpha(Z) = \mathbb{E}\left[Z \mid Z \geq \mathrm{VaR}_\alpha(Z)\right], \tag{34}$$

where $\mathrm{VaR}_\alpha(Z)$ is the Value-at-Risk at confidence level α. Unlike chance constraints, CVaR explicitly captures the magnitude of rare but potentially catastrophic outcomes. Within this risk-measure-based perspective, Yin *et al.* [60] introduced Risk-Aware MPPI (RA-MPPI) by incorporating CVaR as a risk metric within the MPPI framework, thereby enabling tail-risk-sensitive motion planning. In contrast, Parwana *et al.* [61] proposed a risk-aware MPPI framework for stochastic hybrid systems based on confidence-bound formulations, in which constraint-violation risk is penalized through a lower confidence bound of the safety margin. This formulation avoids MC sampling while maintaining comparable probabilistic guarantees, enabling real-time implementation.

3.2.3. Safety Filtering

While the previous subsection focuses on regulating safety in a probabilistic or risk-aware manner, safety filtering aims to enforce hard safety constraints directly at the control input level. The most representative methodology for implementing a safety filter is the use of Control Barrier Functions (CBFs). A CBF defines a safe set within the state space and specifies conditions on the control input to prevent the system from exiting this set. Assuming the system dynamics are affine with respect to the control input, such as $\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + G(\boldsymbol{x})\boldsymbol{u}$, and the safe set is defined as $C = \{\boldsymbol{x} \in \mathbb{R}^n | h(\boldsymbol{x}) \geq 0\}$, the CBF condition to ensure forward invariance is expressed as:

$$\frac{\partial h}{\partial \boldsymbol{x}}\big(f(\boldsymbol{x}) + G(\boldsymbol{x})\boldsymbol{u}\big) \;\geq\; -\alpha\big(h(\boldsymbol{x})\big), \tag{35}$$

where α is an extended class $\mathcal{K}$ function that specifies the decay rate of the safety margin. This inequality becomes a linear constraint on the control input u and can be efficiently integrated into optimization-based control frameworks. While CBFs naturally fit into QP formulations used in many MPC schemes, their integration with sampling-based methods like MPPI is less straightforward. Since MPPI generates control sequences through stochastic sampling rather than solving an optimization problem with explicit constraints, incorporating hard CBF constraints requires alternative strategies that either modify the generated samples to satisfy safety conditions or reshape the sampling process itself to produce only safe trajectories.

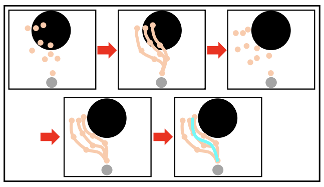
Among such approaches, Yin *et al.* conducted pioneering work with Shield-MPPI [62]. This method augments the MPPI objective with a discrete-time CBF-based penalty to discourage unsafe rollouts during sampling. When the control computed by MPPI is likely to violate the safety constraint, the method further applies a local gradient-based repair step to satisfy the CBF condition. However, because the safety penalty is applied only over a finite horizon and the repair operation remains local, this design may still suffer from performance degradation and suboptimal long-term safety behavior in more challenging scenarios. To address these limitations, the same research group subsequently developed Neural Shield-Variational Inference MPC (NS-VIMPC) [63]. Instead of relying solely on horizon-limited penalties and local repair, this approach learns an approximate discrete-time CBF using a neural network and integrates it with a resampling-based rollout strategy that actively reshapes the sampling process to retain only safe trajectories. By combining learned barrier functions with sampling-aware safety enforcement, NS-VIMPC significantly improves long-term safety performance while maintaining high computational efficiency.

Bridging the gap between post-hoc safety penalties and fully by-construction safety filtering, Zhao *et al.* [64] proposed Lyapunov Function augmented-MPPI (LF-MPPI), which integrates stability and safety enforcement across both the cost and execution stages. In LF-MPPI, Control Lyapunov Functions (CLFs) are first incorporated into the MPPI objective as soft penalties to suppress the influence of unstable trajectories sampled during the rollout. After the standard MPPI update produces a control input $\boldsymbol{u}_t^*$ as given in (30), a QP is subsequently solved at the execution stage to enforce CBF constraints, yielding a safety-filtered control input. Through this two-stage design, LF-MPPI combines penalty-based stabilization during sampling with hard constraint enforcement at the input level. While this hybrid formulation significantly improves closed-loop stability and safety compared to purely penalty-based approaches, it requires solving a QP at every control step, which introduces additional computational overhead and motivates further simplification toward closed-form safety filtering methods. The methods discussed above focus on penalizing or repairing potentially unsafe trajectories after generation. In contrast, proactive approaches enforce safety by construction, modifying the realized control input so that only safe trajectories are sampled and executed. Rabiee and Hoagg [65] proposed Guaranteed-Safe MPPI (GS-MPPI), which integrates multiple safety constraints into a single composite CBF and derives a closed-form minimum-intervention safety filter. Specifically, the authors define a safety mapping function that transforms any desired control input u (encompassing both noisy samples during rollout and the final optimized input) into a guaranteed-safe input $\boldsymbol{u}^{\text{safe}}$:

$$\boldsymbol{u}^{\text{safe}}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{u} + \frac{L_g h(\boldsymbol{x})^\top \max\{0, -\omega(\boldsymbol{x}, \boldsymbol{u}, 0)\}}{L_g h(\boldsymbol{x})\, L_g h(\boldsymbol{x})^\top + \gamma^{-1} h(\boldsymbol{x})^2}, \tag{36}$$

where $L_g h(\boldsymbol{x})$ denotes the Lie derivative of the composite CBF along the control vector field, $\gamma > 0$ regulates the intervention level, and $\omega(\boldsymbol{x}, \boldsymbol{u}, 0) \triangleq L_f h(\boldsymbol{x}) + L_g h(\boldsymbol{x})\boldsymbol{u} + \alpha(h(\boldsymbol{x}))$ represents the Relaxed-CBF safety constraint function. By embedding this closed-form filter directly into the system dynamics used by MPPI, GS-MPPI ensures that every sampled trajectory satisfies safety constraints by construction, while the same filter is applied again at the execution stage to guarantee safety against model uncertainties.

Extending this proactive paradigm to stochastic systems, Tao *et al.* [66] introduced SCBF-MPPI, which incorporates Stochastic Control Barrier Functions (SCBFs) into the MPPI framework at the sampling-distribution level. Instead of filtering or repairing unsafe trajectories after sampling, SCBF-MPPI directly constrains the mean and covariance of the sampling distribution to satisfy SCBF-based probabilistic safety conditions. By proactively restricting the sampling region to probabilistically safe subsets of the control space, SCBF-MPPI guarantees safety in a stochastic sense while significantly improving sample efficiency.


(a)

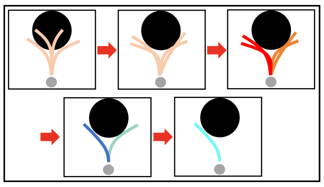

(b)

Figure 6: Methodological comparison of two post-sampling correction strategies. (a) SCP-MPPI [69] transports sampled trajectories toward low-cost regions using SVGD, compensating for the low predictive resolution of sparse control points. (b) CSC-MPPI [70] moves samples toward feasible regions via primal–dual gradient updates, enforcing KKT-based constraints before selecting representative feasible trajectories.

Beyond the CBF paradigm, researchers have explored alternative mathematical foundations for safety filters to address specific limitations or application requirements. Borquez *et al.* [67] developed DualGuard MPPI, which utilizes Hamilton-Jacobi Reachability Analysis to preemptively block the generation of unsafe trajectories during the sampling process while also applying a least-restrictive safety filter at the output. This approach offers safety certificates grounded in differential games rather than barrier inequalities. Wang et al. [68] proposed DBaS-Log-MPPI, which embeds safety directly into the system dynamics via discrete barrier states, transforming hard constraints into smooth state-dependent costs. This design mitigates the sampling inefficiency and penalty tuning challenges typical of CBF-augmented approaches, achieving proactive safety and balanced exploration in cluttered environments.

3.2.4. Constraint-Aware Sampling

The standard Gaussian sampling strategy employed in MPPI does not explicitly account for control or environmental constraints, leading to substantial sampling inefficiency when a large portion of generated trajectories violate dynamic limits or collide with obstacles. Since such infeasible samples receive negligible importance weights, the quality of the resulting update can be significantly degraded, particularly in tightly constrained environments.

To address this limitation, a line of recent studies has introduced post-sampling correction mechanisms that explicitly enforce feasibility by modifying sampled control sequences before trajectory rollouts. Within this category, Andrejev *et al.* [71] proposed π-MPPI, which integrates a projection-based filter to enforce control magnitude and higher-order derivative constraints. Each sampled control sequence is minimally perturbed to satisfy predefined bounds through a quadratic programming-based projection step. This projection process is efficiently parallelized on GPUs using a customized Alternating Direction Method of Multipliers (ADMM) solver with neural-network-based warm starts. However, since the correction operates purely in the control-input space, state-level feasibility, such as collision avoidance, is not directly enforced.

To enhance state-level feasibility, Miura *et al.* [69] proposed Sparse Control Points MPPI (SCP-MPPI), which combines spline interpolation with sample transport. Instead of sampling full control sequences, SCP-MPPI samples sparse control points and reconstructs smooth trajectories via cubic spline interpolation. To compensate for the reduced temporal resolution and mitigate collision risks, SVGD is applied to iteratively transport the sparse control points toward collision-free and low-cost regions while preserving diversity through repulsive kernel interactions.

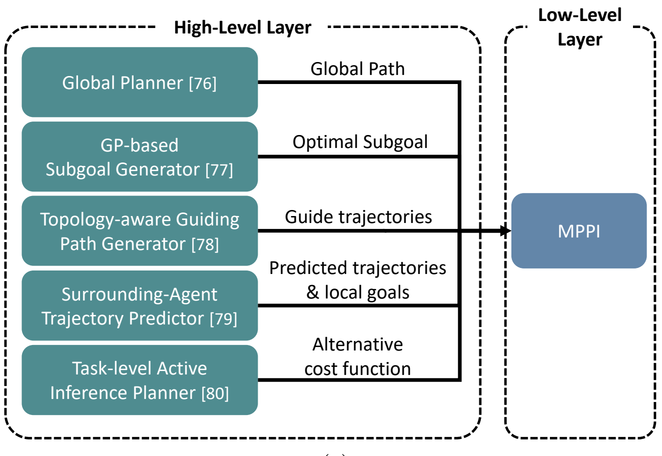
Adopting a different correction paradigm, Park *et al.* [70] proposed Constrained Sampling Cluster-MPPI (CSC-MPPI), which integrates primal–dual constraint refinement with density-based clustering. Each sampled control input is first corrected through iterative primal–dual updates until the Karush–Kuhn–Tucker (KKT) conditions are approximately satisfied. Subsequently, DBSCAN clustering is applied to the feasible trajectories, and representative samples from each cluster are selected for control synthesis. This clustering-based selection mitigates instability that may arise from direct weighted averaging of heterogeneous feasible trajectories. A visual comparison of SCP-MPPI [69] and CSC-MPPI [70] is provided in Fig. 6.

3.3. Framework Design

Recent studies have extended the MPPI control framework by combining it with various forms of optimization [72], learning [73], and inference algorithms [74] to enhance convergence and adaptability. In addition, evolutionary optimization approaches, such as the MPPI–Genetic Algorithm (MPPI-GA) [75], have demonstrated the potential of integrating MPPI with heuristic search frameworks. While such algorithmic combinations are common across stochastic optimal control frameworks, a particularly active research trend focuses on structural integration, where MPPI is embedded within broader control architectures. These architecture-oriented designs aim to leverage the strengths of MPPI in handling nonlinear dynamics while addressing system-level concerns such as scalability and long-horizon planning. The following subsections review representative works that adopt this structural perspective, organized into two lines of integration: hierarchical integration and multi-MPPI integration.

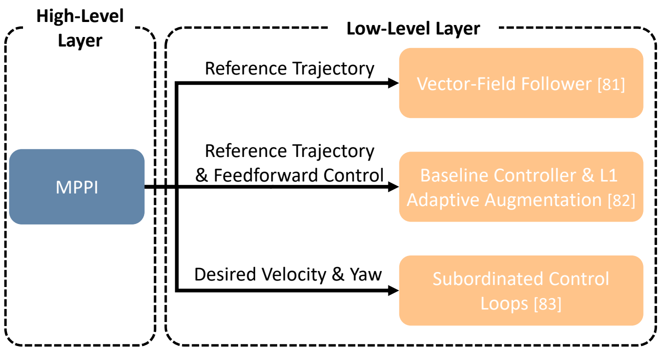
3.3.1. Hierarchical Integration

Hierarchical integration incorporates MPPI within multi-level control architectures that separate high-level planning from low-level execution, addressing the computational complexity of long-horizon trajectory optimization through problem decomposition. MPPI can operate at either level depending on system requirements, serving as a local optimizer that refines coarse plans from higher-level modules or as a trajectory generator tracked by dedicated low-level controllers. This architectural flexibility enables hierarchical systems to balance global coherence with local reactivity while maintaining scalability and modularity.



High-Level Layer
Low-Level Layer
Global Planner [76]
GP-based Subgoal Generator [77]
Topology-aware Guiding Path Generator [78]
Surrounding-Agent Trajectory Predictor [79]
Task-level Active Inference Planner [80]
Global Path
Optimal Subgoal
Guide trajectories
Predicted trajectories & local goals
Alternative cost function
MPPI

(a)



High-Level Layer
Low-Level Layer
MPPI
Reference Trajectory
Reference Trajectory & Feedforward Control
Desired Velocity & Yaw
Vector-Field Follower [81]
Baseline Controller & L1 Adaptive Augmentation [82]
Subordinated Control Loops [83]

(b)

Figure 7: Overview of hierarchical integration architectures. (a) MPPI serves as a low-level local optimizer that refines coarse plans or guides provided by high-level planners. (b) MPPI functions as a high-level trajectory generator, providing reference states to dedicated low-level tracking controllers.

A primary category within this architecture employs MPPI as a local trajectory refiner that executes or optimizes a coarse plan generated by a higher-level module. For instance, Li *et al.* [76] proposed a Global Path Guided MPPI framework in which a high-level global planner provides coarse trajectories, while MPPI refines them at the local level. This integration prevents MPPI from falling into local minima and enables scalability in large-scale navigation problems, though it inevitably depends on the accuracy of the global planner. Similarly, Mohamed *et al.* [77] introduced a Gaussian Process-guided MPPI (GP-guided MPPI), where Sparse Gaussian Processes are employed to recommend informative subgoals. In this case, the GP functions as an upper-level reasoning module under uncertainty, while MPPI ensures feasible execution.

Along the same lines, Zhang *et al.* [78] developed the Risk-Aware Parallel Planner (RAPA-Planner), which employs a lightweight front-end module to generate guiding trajectories. The resulting trajectories are then optimized by a back-end Risk-Aware MPPI (RA-MPPI) controller that incorporates the CVAR, as defined in (34), to quantify and mitigate risk. This hierarchical front-end/back-end structure enhances safety but requires careful tuning of risk-related parameters. This concept of hierarchical planning also extends to the prediction and interaction level. Jansma *et al.* [79] proposed an interaction-aware hierarchical framework that integrates a learning-based trajectory prediction module with the IA-MPPI controller. The upper-level predictor estimates local goals of surrounding agents, while the lower-level MPPI refines motion plans in a decentralized manner, enabling communication-free, interaction-aware navigation in dense multi-agent environments. Furthermore, hierarchical integration can be extended to the task–motion level. Zhang *et al.* [80] proposed a framework combining an Active Inference planner (AIP) with a novel Multi-Modal Model Predictive Path Integral controller (M3P2I). In this system, the task-level AIP generates alternative cost functions corresponding to different actions, while the M3P2I controller samples across them to realize motion-level execution. This dual-layer approach supports reactive switching between tasks such as pushing, pulling, or grasping, though the additional sampling modes increase the computational burden.

A second category of hierarchical design reverses this relationship, employing MPPI as a high-level trajectory generator whose output is tracked by a separate, often faster, low-level controller. This approach leverages MPPI for its planning strengths while delegating real-time safety and robustness to a dedicated follower. Vasilopoulos *et al.* [81] introduced the Reactive Action and Motion Planner (RAMP), where MPPI generates candidate trajectories that are tracked by a low-level vector-field follower. This arrangement ensures safety and reactivity, although it can compromise global optimality when the follower deviates from optimized plans. Another line of hierarchical integration involves adaptive and subordinate control architectures. Pravitra et al. [82] proposed an L1-Adaptive MPPI, where MPPI serves as a nominal trajectory generator and an L1 adaptive controller provides real-time tracking compensation. This structure enhances the hierarchical separation between trajectory optimization and feedback adaptation, albeit with increased complexity. Homburger et al. [83] further investigated subordinated control loops with MPPI, explicitly evaluating how a high-level MPPI interacts with classical feedback controllers. The analysis shows that such layering establishes a clear interface between planning and low-level control, exemplifying the modularity of hierarchical MPPI systems. To visualize the distinction between these two hierarchical paradigms, we present a schematic overview in Fig. 7, highlighting the input-output relationships specific to each configuration.

3.3.2. Multi-MPPI Integration

Beyond these hierarchical integrations between MPPI and external planners or controllers, recent research extends the MPPI framework by employing multiple MPPI instances within a unified control structure to achieve more diverse behaviors and improved overall performance.

For instance, Zhou et al. [84] developed the Parallel MPPI (PMPPI) architecture, in which multiple planners with different objectives such as goal seeking and obstacle avoidance run concurrently. A separate Judge strategy evaluates their outputs and fuses them into a single control command. This structural decomposition allows diverse behaviors to be coordinated under one unified MPPI framework, improving robustness in dynamic environments but increasing computational load due to multi-instance execution. Similarly, Raisi et al. [85] proposed the Planner–Estimator MPPI (PE-MPPI) for fault-tolerant control. Two MPPIs operate in parallel. The Planner MPPI generates control actions based on the nominal model, while the Estimator MPPI infers model parameters when deviations between the predicted and actual system behaviors occur. This cooperative dual structure continuously updates the dynamics model of the planner and maintains control stability under actuator or parameter faults. Jung et al. [86] further evolved this multi-instance concept into a nested framework known as Contingency-MPPI. In this architecture, an outer nominal MPPI optimizes the primary trajectory, while an inner contingency MPPI is executed at every sampled state to verify the existence of a valid backup path leading to a safe zone. If the inner planner fails to find a feasible contingency plan for any state along a nominal trajectory, that trajectory is immediately discarded. This rigorous nested design guarantees real-time safety assurance, albeit at the cost of significantly increased sampling requirements.

3.4. Robustness Enhancement

Real-world robotic systems seldom operate in benign conditions. Instead, they are constantly challenged by model inaccuracies, exogenous disturbances, and incomplete or noisy information about the task environment. While the baseline MPPI algorithm has demonstrated remarkable performance in nominal conditions, its risk-neutral design renders it fragile in the face of uncertainty and disturbances. As a result, a significant body of work has focused on extending MPPI toward more robust formulations. In this subsection, we review advances along two dimensions: methods that explicitly or implicitly account for uncertainty in dynamics, costs, and sampling distributions, and methods that enhance robustness to external disturbances and environmental variability.

3.4.1. Robustness to Uncertainty

The baseline MPPI formulation relies on a nominal dynamics model and assumes idealized conditions during trajectory evaluation. In practical robotic systems, however, several forms of uncertainty inevitably arise, including parametric mismatch in the system model, stochastic disturbances that affect state evolution, and variability in sensing or actuation. These discrepancies between the nominal model and real-world behavior have motivated many extensions that directly address these uncertainties, allowing MPPI to better handle model errors, external perturbations, and other sources of variability encountered during deployment.

A primary challenge in model-based control is the discrepancy between the mathematical model and the physical system. To address parameter mismatch, Abraham *et al.* [87] extended the free-energy formulation in (5) to explicitly incorporate uncertainty over model parameters. In their approach, the system dynamics are parameterized by a distribution $p(\bar{\theta})$, and the free energy is defined as

$$F_\lambda = -\lambda \log \mathbb{E}_{p(\bar{\theta}),\mathbb{P}}\left[\exp\left(-\frac{1}{\lambda}S(\boldsymbol{V})\right)\right]. \tag{37}$$

This expression expands the expectation in the nominal formulation to capture both uncertainty in the model parameters and the stochasticity introduced by trajectory sampling. The resulting formulation allows trajectories to be evaluated under a broader uncertainty model without requiring explicit identification of the underlying parameters. Addressing similar parametric sensitivities, Nyboe *et al.* [88] proposed a robust MPPI formulation based on embedded sensitivity tubes. They propagate state–parameter sensitivities alongside the nominal dynamics to approximate local uncertainty envelopes. These tubes are incorporated into the cost function as robustness penalties, enabling stable control under parameter mismatch. Alternatively, instead of explicit parameterization, Cong *et al.* [89] proposed Recurrent MPPI (RMPPI), which employs a recurrent neural network trained under domain randomization. The LSTM model adapts online to compensate for unmodeled physical parameters and distributional shifts, demonstrating that robustness can emerge from data-driven adaptation.

Closely related to parameter mismatch is the epistemic uncertainty inherent in learned dynamics models, particularly in data-sparse regions. Arruda *et al.* [90] addressed this by incorporating predictive variance from learned forward models into the running cost. Accordingly, the trajectory cost in (3) is reformulated to explicitly account for predictive uncertainty as

$$S(\boldsymbol{V}) = \Phi(\boldsymbol{x}_T) + \sum_{t=0}^{T-1} \ell(\boldsymbol{x}_t, \hat{\sigma}_t), \tag{38}$$

where $\hat{\sigma}_t$ denotes the predictive variance obtained from a Gaussian process or a mixture-density network. For instance, in a goal-tracking task, the uncertainty-aware running cost can be written as

$$\ell(\boldsymbol{x}_t, \hat{\sigma}_t) = \gamma\hat{\sigma}_t + \big(\boldsymbol{x}_t - \boldsymbol{x}_{\text{goal}}\big)^\top Q\big(\boldsymbol{x}_t - \boldsymbol{x}_{\text{goal}}\big). \tag{39}$$

This risk-sensitive formulation encourages trajectories that avoid high-uncertainty regions. Similarly, Kalaria *et al.* [91] developed an uncertainty-aware MPPI that estimates ensemble-based predictive variance and adds it as a risk-sensitive term in the cost. Coupled with meta-learning–driven model adaptation, this method achieves

Table 2: Uncertainty types addressed by MPPI variants.

Method	Parametric	Process Noise	Epistemic	Execution Error	Environment
Abraham et al. [87]	✓				
Nyboe et al. [88]	✓				
Cong et al. [89]	✓				
Arruda et al. [90]	✓		✓		
Kalaria et al. [91]	✓		✓		
Wang et al. [92]		✓			
Mohamed et al. [93]		✓			
Higgins et al. [94]				✓	
Patrick & Bakolas [95]					✓

✓: Addressed

robust performance under changing dynamics and surface conditions. Beyond the structure of the model, the nature of the stochastic process itself can also introduce additional uncertainty, particularly when the Gaussian noise assumption in (13) is violated. To address this limitation, Wang *et al.* [92] extended the baseline dynamics in (13) to a jump-diffusion system in order to capture rare but significant stochastic shocks. The resulting extended dynamics are expressed as

$$d\boldsymbol{x}_t = f(\boldsymbol{x}_t)\,dt + G(\boldsymbol{x}_t)\,\boldsymbol{u}_t\,dt + B(\boldsymbol{x}_t)\,d\boldsymbol{w}_t + H(\boldsymbol{x}_t)\,d\boldsymbol{P}_t, \quad (40)$$

where $H(\boldsymbol{x}_t)\,d\boldsymbol{P}_t$ introduces discontinuous jumps governed by a Poisson process. This generalization enables the modeling of non-Gaussian disturbances commonly observed in contact-rich conditions. In a related effort to handle stochastic state evolution, Mohamed *et al.* [93] proposed the Unscented MPPI (U-MPPI). This approach leverages the Unscented Transform (UT) to propagate both the mean and covariance of the state distribution through nonlinear dynamics. By jointly considering uncertainty propagation and introducing a risk-sensitive cost that penalizes high predictive variance, U-MPPI achieves safer control under stochastic uncertainty.

Uncertainty also arises downstream at the execution level, where the low-level controller may fail to perfectly track the high-level plan. Higgins *et al.* [94] addressed this by embedding a data-informed risk term into the MPPI cost for UAV planning. In their approach, tracking errors are modeled as collision risk distributions dependent on vehicle speed. By penalizing states with high risk probability, the method enables the UAV to proactively adjust its velocity near obstacles, effectively transforming unpredictable tracking errors into a quantitative measure of uncertainty within the planner.

Finally, uncertainty exists in the external environment and the sampling process itself. Patrick and Bakolas [95] addressed the risk of multi-modal cost landscapes and dynamic environments. They utilized rollout clustering to prevent the weighted average of trajectories from falling into unsafe regions and modeled dynamic-obstacle motion probabilistically within each rollout. This allows the controller to anticipate stochastic obstacle behavior and generate collision-averse trajectories in nonstationary environments. To consolidate the above literature, Table 2 summarizes the types of uncertainty addressed by each MPPI variant.

3.4.2. Robustness to Disturbances

While uncertainty in system parameters and models is a critical challenge, disturbances constitute an equally important factor that can severely compromise the performance of MPPI. Real-world systems are exposed to sudden shocks, structured external forces, and adversarial environmental effects that cannot be captured by Gaussian input noise alone. To address this, several extensions of MPPI have been proposed that explicitly focus on disturbance rejection and robust performance.

Williams et al. [96] proposed a robust extension of MPPI based on a tube-based model predictive control framework to explicitly handle external disturbances. In their approach, MPPI optimizes a nominal trajectory using a disturbance-free system model, while an ancillary tracking controller stabilizes the deviation between the nominal and the actual disturbed system. This tube-based separation ensures that the real system state remains within a bounded neighborhood of the nominal MPPI trajectory, thereby providing robustness against additive disturbances and model perturbations. Building on this, Gandhi et al. [97] generalized the approach into Robust-MPPI (RMPPI) by establishing a rigorous theoretical foundation for disturbance rejection. RMPPI formalizes robustness through contraction-metric based tracking and derives explicit robustness guarantees via free-energy growth bounds. This work marks a critical transition from empirically validated robust behavior to a theoretically guaranteed robust MPPI framework.

Liang et al. [98] further applied MPPI to missile guidance under adversarial conditions such as maneuvering targets, varying interceptor velocity, and actuator failures. Their framework integrates a deep neural network dynamics model into MPPI and employs meta-learning for rapid online adaptation, thereby compensating for environmental perturbations and hardware faults. By adapting the learned dynamics to actuator loss-of-effectiveness and target maneuvers in real-time, the method achieves robust interception with desired terminal impact angles, demonstrating MPPI's applicability to safety-critical,

Table 3: Summary of MPPI implementation approach and their characteristics.

Implementation Approach	Compute Mechanism	Main Advantages	Main Limitations
Pure CPU (Section. 4.1.1)	Sequential CPU execution	Low cost, Easy prototyping, Embedded compatibility	Limited parallelism, Low frequency, Poor scalability
GPU/CUDA (Section. 4.1.2)	Custom CUDA kernels	Thread-level parallelism, Manual memory optimization	Hardware dependence, High development complexity
High-level GPU frameworks (Section. 4.1.3)	Tensor vectorization	Modularity, Autodiff, Learning integration	Abstraction overhead, Reduced low-level control
Physics simulator integration (Section. 4.1.4)	Parallel physics stepping	Realistic contact modeling, Physical validity	High computational cost, Simulator fidelity limits

disturbance-rich domains. Transitioning from internal faults to external environmental disturbances, Folk et al. [99] proposed an energy-aware MPPI for UAV navigation in urban environments where gusts act as strong exogenous disturbances. Their method augments the cost function with power consumption penalties informed by onboard LiDAR-based wind predictions, achieving safer trajectories with up to 30% lower energy consumption and fewer collisions, though performance inherently relies on the fidelity of local wind estimation.

4. Real-Time Implementation Strategies and Modular Frameworks for MPPI

MPPI control has evolved from a theoretical stochastic control formulation into a practical real-time framework, enabled by advancements in computational hardware and open-source software. With the rise of GPU acceleration and differentiable simulators, MPPI has transitioned from an isolated algorithmic controller into a shared research baseline that supports reproducibility, scalability, and flexible algorithmic development. This section reviews the evolution of MPPI implementations and introduces a modular open framework designed to facilitate rapid research and real-time deployment.

4.1. Evolution of MPPI Implementations

At each control cycle, MPPI generates thousands of control sequence samples and propagates them through system dynamics to evaluate the expected cost. Because this procedure is computationally intensive, the historical development of MPPI implementations can be viewed as a continuous pursuit of greater numerical efficiency and real-time capability. A concise overview of this evolution is summarized in Table 3, which organizes the major implementation approaches ranging from pure CPU execution to GPU/CUDA kernels, high-level GPU frameworks, and physics-simulator integration, and highlights their computational mechanisms, advantages, and limitations.

4.1.1. Sequential Rollout and Early CPU Implementations

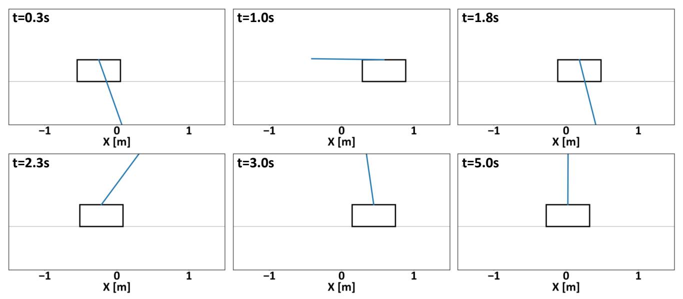
Initial implementations of MPPI were carried out on Central Processing Unit (CPU) using sequential rollout simulations. This configuration reflected the early emphasis of the algorithm on conceptual clarity and theoretical validation rather than execution speed. CPU-based systems offered practical advantages for early experimentation: they required no dedicated hardware, were relatively inexpensive, and could be deployed directly on embedded processors already integrated into robotic platforms. They also provided a deterministic execution environment and mature development tools, which facilitated debugging and algorithmic analysis. However, as the sampling requirements of MPPI increased, CPUs became a fundamental bottleneck. Because each trajectory had to be simulated sequentially, the total computational cost scaled linearly with the number of samples, limiting control frequencies to a few hertz. In practice, high-dimensional or fast-dynamics systems often required simplifying the dynamics model or reducing the number of samples, which degraded control performance. This limitation revealed the intrinsic dependence of MPPI on large-scale parallelization and motivated the transition to GPU-based implementations.

4.1.2. Parallel Sampling on GPU and CUDA Acceleration

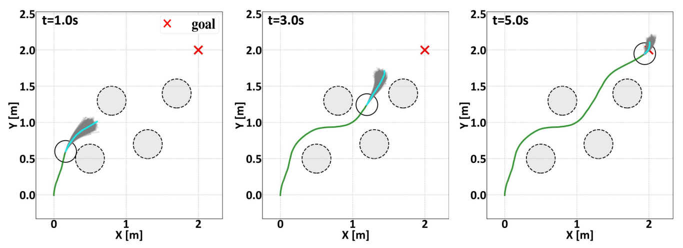
The introduction of GPU enabled MPPI to fully exploit the parallel structure of its sampling process. Each rollout, being independent from others, mapped naturally onto GPU thread-level parallelism, allowing thousands of trajectories to be propagated simultaneously [25]. Through the use of CUDA kernels, MPPI achieved control frequencies of several hundred hertz while maintaining high sampling resolution, thereby enabling MPPI to operate as a real-time stochastic model predictive controller in high-frequency settings [100]. Following this trend, MPPI gradually evolved from a simulation-oriented algorithm into a deployable control framework for high-speed robotic systems. Nevertheless, CUDA-level implementations required manual management of GPU memory, thread scheduling, and kernel fusion, which made them hardware-dependent and difficult to generalize. The resulting systems were fast but rigid, motivating the search for higher-level abstractions that preserved performance while improving code modularity and portability.

4.1.3. High-Level GPU Computing Frameworks

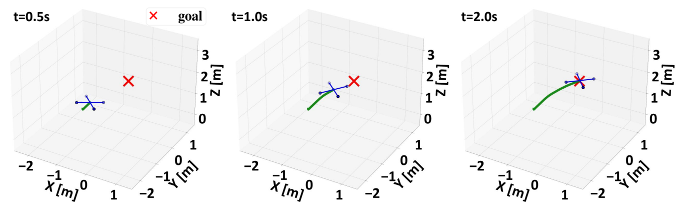
The emergence of high-level GPU computing and differentiable programming frameworks such as JAX and PyTorch marked an important transition in the development of MPPI [16, 70]. These frameworks removed the need for low-level GPU programming while preserv-



(a)



(b)



(c)

Figure 8: Application of the proposed modular framework. (a) Cartpole. (b) Mobile robot. (c) Quadrotor.

ing efficient parallel execution, automatic differentiation, Just-In-Time (JIT) compilation, and large-scale vectorization. Such capabilities allowed MPPI computations, including sampling, cost evaluation, and control updates, to be expressed as vectorized operations applied to trajectory batches. This formulation increased the modularity of the algorithm and simplified performance optimization.

JAX adopts a functional programming style combined with a compilation-based execution model. MPPI computations can be represented as collections of pure and composable functions within this structure, enabling extensive compiler optimizations and predictable execution behavior. These characteristics are advantageous for large-scale stochastic control problems that require both reproducibility and computational efficiency. However, the compilation workflow can introduce overhead when model structures or control logic must be modified during runtime, reducing flexibility in interactive control scenarios.

PyTorch follows a different design philosophy based on a dynamic computation graph constructed during execution. This mechanism supports intuitive debugging and rapid prototyping, particularly in exploratory research or interactive environments. In addition, integration with deep learning software ecosystems allows MPPI implementations built on PyTorch to interact effectively with neural policy networks and data-driven cost models.

4.1.4. Integration with Physics Simulators

The recent generation of MPPI frameworks extends beyond analytical or kinematic models by directly coupling with high-fidelity physics simulators such as MuJoCo, Isaac Gym, and MJX [53, 101, 102, 103]. Through these environments, each sampled trajectory can be propagated under physically realistic conditions that include contact dynamics, frictional interactions, and external disturbances. This integration allows the controller to evaluate not only smooth trajectory optimization but also the feasibility of actions in contact-rich or dynamic environments where nonlinearity and discontinuity play a dominant role. By incorporating real-time physics feedback, MPPI can estimate quantities such as contact forces, impact impulses, and joint constraints during each rollout, leading to more robust and physically grounded control performance. However, the computational cost of this approach is significantly higher than that of purely mathematical cost evaluations, since the underlying physics engines must solve differential contact constraints and collision responses at every simulation step.

4.2. Modular Open Framework

To support extensible research and reproducible evaluation, the proposed open-source MPPI framework provides a minimal yet flexible base implementation rather than a fully featured control suite. Fig. 8 presents three base environments included within the framework for algorithm development and experimentation, which consist of a cartpole system, a two-dimensional mobile robot, and a three-dimensional quadrotor.

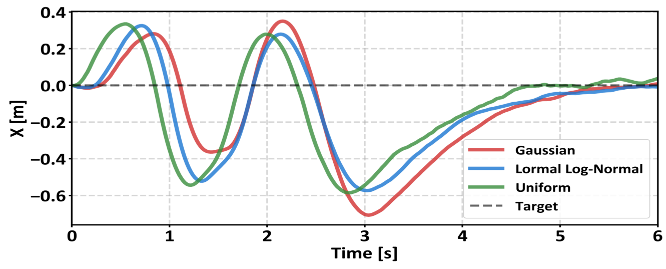
The framework is organized around four core modules: a sampling module for generating stochastic control perturbations, a dynamics module for propagating state trajectories through analytical or simulator-based models, a cost module for evaluating running and terminal costs, and a logging and visualization module for recording simulation data and performance statistics. Table. 4 summarizes these modules and outlines extensibility options that allow researchers to modify or replace individual components without affecting the overall MPPI loop. Researchers may introduce new sampling distributions by subclassing the sampling interface and redefining the rollout generation routine, apply task-specific dynamics models by extending the dynamics module, or incorporate additional cost terms and hierarchical updates through custom cost modules. This modular organization supports the integration of new algorithmic ideas while preserving compatibility with the core MPPI structure.

To facilitate integration with robotic systems, the framework includes a lightweight example interface for ROS 2 Humble. In this example setup, a ROS 2 node subscribes to robot state topics and publishes control commands from the MPPI controller in real time. Researchers may adapt

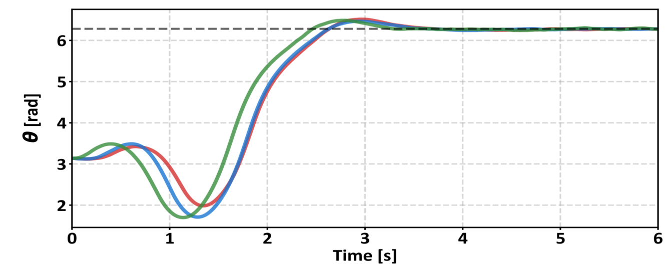
Table 4: Modular structure of the baseline MPPI framework and examples of researchers extensions.

Module	**Function**	**Extensibility**
Sampling Module	Generates stochastic control perturbations (noise) for trajectory rollouts	Implement various distributions, learned sampling priors
Dynamics Module	Propagates state trajectories forward in time using analytical or simulator-based models	Incorporate various robot dynamics, perform parallel physics simulation for rollouts, or replace with learned neural dynamics
Cost Module	Computes state-dependent running costs and terminal costs for performance evaluation	Design custom objectives such as collision avoidance penalties, safety barrier functions, or energy efficiency terms
Logging / Visualization Module	Records simulation data, cost convergence trends, and control performance	Use MuJoCo viewer for real-time visual feedback or Python-based plotting tools for post-analysis

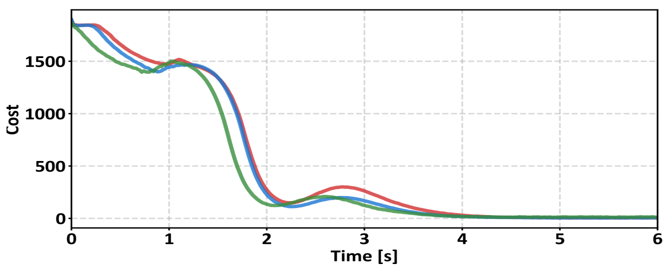
or extend this interface by adding sensory inputs, custom estimators, or coordination nodes, allowing the same MPPI structure to be connected to a wide range of robotic platforms from simple simulators to complex real-world systems without modification to the underlying algorithm.



(a)



(b)



(c)

Figure 9: Optimization results for the Cart-Pole task utilizing interchangeable sampling modules within the proposed framework (Gaussian [24], NLN [38], and Uniform [39]). (a) Cart position x. (b) Pole angle θ. (c) Cost convergence history.

4.2.1. Evaluation of the Framework's Modular Structure

To validate the modularity and correctness of the proposed framework, we evaluate the baseline implementation on the Cart-Pole stabilization task while swapping the sampling module. Three sampling strategies are tested: Gaussian noise, NLN noise, and Uniform noise. Each strategy is integrated into the same MPPI controller without modifying any other component of the framework, demonstrating that sampling distributions can be interchanged seamlessly. Although this evaluation focuses on the sampling module, the experiment reflects the general modular design principle of the framework, where individual components can be interchanged without affecting the core MPPI loop. Fig. 9 shows the resulting trajectories for the cart position x, the pole angle θ, and the cost convergence profile. Across all sampling types, the controller successfully stabilizes the system, confirming that the baseline implementation provides a consistent and reliable foundation for MPPI experiments. Although the transient responses differ slightly due to differences in exploration characteristics, all methods converge to comparable performance. These results demonstrate that the proposed framework provides a fair and reproducible environment for evaluating interchangeable module designs.

4.2.2. Significance of the Framework for Research

The proposed framework is not intended as a full benchmark suite but as a minimal and reproducible baseline that supports research and development activities. It offers a transparent implementation of the core MPPI algorithm together with a consistent structural template that researchers can extend or modify according to their objectives. This modular design enables systematic comparison of algorithmic variations while reducing the engineering effort required to construct low-level control infrastructure.

Because the framework maintains standardized interfaces across the modules, new ideas such as non-Gaussian sampling strategies, neural or learned dynamics models, and cost formulations informed by high-fidelity physics engines like MJX can be evaluated within the same control loop. This consistency allows researchers to isolate the effects of individual design choices without interference from environmental or implementation differences. Through this modular yet minimal composition, the framework serves as a shared experimental foundation for developing and validating advanced MPPI variants. Rather than enforcing a single optimal configuration, it promotes transparent comparison, reproducibility, and cumulative progress in MPPI.

5. Conclusion

This paper presented a comprehensive analysis and implementation framework for MPPI control, a sampling-based stochastic optimal control method. A unified theoretical foundation was first established by connecting information-theoretic interpretations with classical path-integral derivations. Building on this foundation, previously fragmented research was reorganized into four themes: sampling strategies, constraint handling, framework design, and robustness enhancement. This analysis clarified the strengths and limitations of each methodological direction, their structural relationships, and the evolutionary trajectory of MPPI in robotic control. Beyond theoretical investigation, a modular and extensible MPPI framework was developed to support reproducible experimentation and rapid validation of algorithmic variants. By providing a consistent structure and standardized interfaces, it serves as a practical benchmark for developing, comparing, and deploying MPPI-based control algorithms in real robotic systems.

At the same time, our analysis highlights several limitations that must be addressed for MPPI to progress further. Despite recent advances, challenges remain in establishing rigorous theoretical guarantees and practical robustness. Future work must develop formal assurances of stability, convergence, and robustness under finite-sample conditions, particularly for safety-critical applications. Another key direction is the principled integration of learning and control, such as using generative models to construct expressive, context-aware sampling distributions. Progress will also require moving beyond soft-penalty formulations toward more explicit constraint handling, as well as incorporating data-driven dynamics models with uncertainty quantification to reduce the simulation-to-reality gap and improve adaptability under unpredictable disturbances.

Ultimately, the future of MPPI lies in its evolution from a practical sampling-based controller into a unified platform for real-time stochastic decision-making. With this development, MPPI is poised to become a key tool for addressing increasingly complex and demanding problems in robotics.

Acknowledgments

This research was supported by the National Research Foundation of Korea (NRF) (No. RS-2024-00411007), the Ministry of Trade, Industry & Energy (MOTIE) (No. RS-2025-25449039), the Council of Science & Technology (NST) (No. GTL25041-000), the Korea Basic Science Institute (National research Facilities and Equipment Center) (No. RS-2025-00564593), and the KIST Institutional Program (No. 2E33591, 2E33592).

CRediT authorship contribution statement

Leesai Park: Conceptualization, Methodology, Writing - original draft. **Keunwoo Jang:** Methodology, Writing - review & editing. **Sanghyun Kim:** Software, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The simulation framework, experimental data, and implementation code supporting the findings of this study are publicly available at `https://github.com/rcilab/rci_mppi_framework`.

References

[1] B. Kouvaritakis, M. Cannon, Model predictive control, Switzerland: Springer International Publishing 38 (13-56) (2016) 7.

[2] M. Toussaint, J. Harris, J.-S. Ha, D. Driess, W. Hönig, Sequence-of-constraints mpc: Reactive timing-optimal control of sequential manipulation, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2022, pp. 13753–13760.

[3] O. Villarreal, V. Barasuol, P. M. Wensing, D. G. Caldwell, C. Semini, Mpc-based controller with terrain insight for dynamic legged locomotion, in: 2020 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2020, pp. 2436–2442.

[4] M. Elobaid, G. Turrisi, L. Rapetti, G. Romualdi, S. Dafarra, T. Kawakami, T. Chaki, T. Yoshiike, C. Semini, D. Pucci, Adaptive non-linear centroidal mpc with stability guarantees for robust locomotion of legged robots, IEEE Robotics and Automation Letters (2025).

[5] D. Benders, J. Köhler, T. Niesten, R. Babuška, J. Alonso-Mora, L. Ferranti, Embedded hierarchical mpc for autonomous navigation, IEEE Transactions on Robotics (2025).

[6] S. Dixit, U. Montanaro, M. Dianati, D. Oxtoby, T. Mizutani, A. Mouzakitis, S. Fallah, Trajectory planning for autonomous high-speed overtaking in structured environments using robust mpc, IEEE Transactions on Intelligent Transportation Systems 21 (6) (2019) 2310–2323.

[7] M. Diehl, H. G. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, F. Allgöwer, Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations, Journal of Process Control 12 (4) (2002) 577–585.

[8] J. Wang, S. Kim, T. S. Lembono, W. Du, J. Shim, S. Samadi, K. Wang, V. Ivan, S. Calinon, S. Vijayakumar, et al., Online multicontact receding horizon planning via value function approximation, IEEE Transactions on Robotics 40 (2024) 2791–2810.

[9] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, S. Schaal, Stomp: Stochastic trajectory optimization for motion planning, in: 2011 IEEE international conference on robotics and automation, IEEE, 2011, pp. 4569–4574.

[10] M. Kobilarov, Cross-entropy randomized motion planning, in: Robotics: Science and Systems, Vol. 7, MIT Press, 2012, pp. 153–160.

[11] Z. I. Botev, D. P. Kroese, R. Y. Rubinstein, P. L'ecuyer, The cross-entropy method for optimization, in: Handbook of statistics, Vol. 31, Elsevier, 2013, pp. 35–59.

[12] A. Mavrommati, C. Osorio, R. G. Valenti, A. Rajhans, P. J. Mosterman, An application of model predictive control to reactive motion planning of robot manipulators, in: 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), IEEE, 2021, pp. 915–920.

[13] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, E. A. Theodorou, Aggressive driving with model predictive path integral control, in: 2016 IEEE international conference on robotics and automation (ICRA), IEEE, 2016, pp. 1433–1440.

[14] Y. Lee, K. H. Choi, K.-S. Kim, Gpu-enabled parallel trajectory optimization framework for safe motion planning of autonomous vehicles, IEEE Robotics and Automation Letters (2024).

[15] K. Matsui, F. Adan, P. Angeloudis, Decentralised motion planning for autonomous mining haulage trucks using prioritised multi-agent mppi, IFAC-PapersOnLine 58 (10) (2024) 154–161.

[16] M. Bhardwaj, B. Sundaralingam, A. Mousavian, N. D. Ratliff, D. Fox, F. Ramos, B. Boots, Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation, in: Conference on Robot Learning, PMLR, 2022, pp. 750–759.

[17] D. Kim, E. Im, Y. Kim, M. Lim, Y. Lee, Single-instance sampling for computationally efficient and accurate real-time task space mppi control, IEEE Transactions on Robotics 41 (2025) 6327–6344. `doi:10.1109/TRO.2025.3626660`.

[18] I. S. Mohamed, G. Allibert, P. Martinet, Sampling-based mpc for constrained vision based control, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2021, pp. 3753–3758.

[19] K. Vasios, A. Porichis, V. Mohan, P. Chatzakos, Robotic mushroom harvesting with real2sim2real and model predictive path integral (mppi) based planning, in: 2025 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2025, pp. 13131–13137.

[20] M. Minařík, R. Pěnička, V. Vonásek, M. Saska, Model predictive path integral control for agile unmanned aerial vehicles, in: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2024, pp. 13144–13151.

[21] I. S. Mohamed, G. Allibert, P. Martinet, Model predictive path integral control framework for partially observable navigation: A quadrotor case study, in: 2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV), IEEE, 2020, pp. 196–203.

[22] K.-P. Kim, C.-H. Lee, Fixed range horizon mppi-based missile computational guidance for constrained impact angle, International Journal of Control, Automation and Systems 21 (6) (2023) 1866–1884.

[23] K.-P. Kim, C.-H. Lee, Mppi-based computational guidance for impact time control with practical constraints: K.-p. kim, c.-h. lee, International Journal of Aeronautical and Space Sciences 24 (3) (2023) 853–875.

[24] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, E. A. Theodorou, Information-theoretic model predictive control: Theory and applications to autonomous driving, IEEE Transactions on Robotics 34 (6) (2018) 1603–1622.

[25] G. Williams, A. Aldrich, E. A. Theodorou, Model predictive path integral control: From theory to parallel computation, Journal of Guidance, Control, and Dynamics 40 (2) (2017) 344–357.

[26] C. Tao, H. Kim, N. Hovakimyan, Rrt guided model predictive path integral method, arXiv preprint arXiv:2301.13143 (2023).

[27] J. Yang, M. Kang, S. Lee, S. Kim, Hybrid a*-guided model predictive path integral control for robust navigation in rough terrains, Mathematics 13 (5) (2025) 810.

[28] Y. Huang, H. Liu, Diffusion-mppi: Diffusion informed model predictive path integral method, in: International Conference on Neural Information Processing, Springer, 2024, pp. 284–298.

[29] R. Kusumoto, L. Palmieri, M. Spies, A. Csiszar, K. O. Arras, Informed information theoretic model predictive control, in: 2019 International Conference on Robotics and Automation (ICRA), IEEE, 2019, pp. 2047–2053.

[30] Y. Qu, H. Chu, S. Gao, J. Guan, H. Yan, L. Xiao, S. E. Li, J. Duan, Rl-driven mppi: Accelerating online control laws calculation with offline policy, IEEE Transactions on Intelligent Vehicles 9 (2) (2023) 3605–3616.

[31] P. Wang, C. Li, C. Weaver, K. Kawamoto, M. Tomizuka, C. Tang, W. Zhan, Residual-mppi: Online policy customization for continuous control, in: The Thirteenth International Conference on Learning Representations.

[32] K. Honda, N. Akai, K. Suzuki, M. Aoki, H. Hosogaya, H. Okuda, T. Suzuki, Stein variational guided model predictive path integral control: Proposal and experiments with fast maneuvering vehicles, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2024, pp. 7020–7026.

[33] M. Lee, D. Lee, Time-correlated model predictive path integral: Smooth action generation for sampling-based control, in: 2025 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2025, pp. 14490–14496.

[34] B. Vlahov, J. Gibson, D. D. Fan, P. Spieler, A.-a. Agha-mohammadi, E. A. Theodorou, Low frequency sampling in model predictive path integral control, IEEE Robotics and Automation Letters 9 (5) (2024) 4543–4550.

[35] T. Kim, J. Jeon, M.-T. Lim, Y. Lee, Y. Oh, Simultaneous tracking and balancing control of two-wheeled inverted pendulum with roll-joint using dynamic variance mppi, in: 2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids), IEEE, 2024, pp. 417–424.

[36] H. Xue, C. Pan, Z. Yi, G. Qu, G. Shi, Full-order sampling-based mpc for torque-level locomotion control via diffusion-style annealing, in: 2025 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2025, pp. 4974–4981.

[37] D. M. Asmar, R. Senanayake, S. Manuel, M. J. Kochenderfer, Model predictive optimized path integral strategies, in: 2023 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2023, pp. 3182–3188.

[38] I. S. Mohamed, K. Yin, L. Liu, Autonomous navigation of agvs in unknown cluttered environments: log-mppi control strategy, IEEE Robotics and Automation Letters 7 (4) (2022) 10240–10247.

[39] Y. Seo, D. Kim, J. Bak, Y. Oh, Y. Lee, Extremely fast computation of com trajectory generation for walking leveraging mppi algorithm, in: 2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids), IEEE, 2023, pp. 1–7.

[40] O. G. Poyrazoglu, Y. Cao, V. Isler, C-uniform trajectory sampling for fast motion planning, in: 2025 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2025, pp. 9236–9242.

[41] H. Homburger, S. Wirtensohn, M. Diehl, J. Reuter, Feature-based mppi control with applications to maritime systems, Machines 10 (10) (2022) 900.

[42] E. Trevisan, J. Alonso-Mora, Biased-mppi: Informing sampling-based model predictive control by fusing ancillary controllers, IEEE Robotics and Automation Letters 9 (6) (2024) 5871–5878.

[43] L. L. Yan, S. Devasia, Output-sampled model predictive path integral control (o-mppi) for increased efficiency, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2024, pp. 14279–14285.

[44] G. Zhao, N. Jin, J. Wu, Z. Xiong, Online motion generation via tangential sampling-based mpc around nonconvex obstacles, IEEE Robotics and Automation Letters (2025).

[45] M. Aoki, K. Honda, H. Okuda, T. Suzuki, Switching sampling space of model predictive path-integral controller to balance efficiency and safety in 4wids vehicle navigation, in: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2024, pp. 3196–3203.

[46] T. Kim, G. Park, K. Kwak, J. Bae, W. Lee, Smooth model predictive path integral control without smoothing, IEEE Robotics and Automation Letters 7 (4) (2022) 10406–10413.

[47] J. Sacks, B. Boots, Learning sampling distributions for model predictive control, in: Conference on Robot Learning, PMLR, 2023, pp. 1733–1742.

[48] T. Power, D. Berenson, Learning a generalizable trajectory sampling distribution for model predictive control, IEEE Transactions on Robotics 40 (2024) 2111–2127.

[49] T. Fuke, M. Endo, K. Honda, G. Ishigami, Towards local minima-free robotic navigation: Model predictive path integral control via repulsive potential augmentation, in: 2025 IEEE/SICE International Symposium on System Integration (SII), IEEE, 2025, pp. 1112–1117.

[50] T. Fuke, M. Endo, K. Honda, G. Ishigami, Drpa-mppi: Dynamic repulsive potential augmented mppi for reactive navigation in unstructured environments, arXiv preprint arXiv:2503.20134 (2025).

[51] M. Testouri, G. Elghazaly, R. Frank, Towards a safe real-time motion planning framework for autonomous driving systems: A model predictive path integral approach, in: 2023 3rd International Conference on Robotics, Automation and Artificial Intelligence (RAAI), IEEE, 2023, pp. 231–238.

[52] I. Ardiyanto, E. Firmansyah, Eikonal model predictive path integral for risk-aware mobile robot navigation, in: 2025 11th International Conference on Control, Automation and Robotics (ICCAR), IEEE, 2025, pp. 212–218.

[53] P. N. Crestaz, L. de Matteis, E. Chane-Sane, N. Mansard, A. Del Prete, Td-cd-mppi: Temporal-difference constraint-discounted model predictive path integral control (2025).

[54] J. Yin, Z. Zhang, E. Theodorou, P. Tsiotras, Trajectory distribution control for model predictive path integral control using covariance steering, in: 2022 International Conference on Robotics and Automation (ICRA), IEEE, 2022, pp. 1478–1484.

[55] A. Trivedi, S. Prajapati, A. Shirgaonkar, M. Zolotas, T. Padır, Data-driven sampling based stochastic mpc for skid-steer mobile robot navigation, in: 2025 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2025, pp. 16619–16625.

[56] I. M. Balci, E. Bakolas, B. Vlahov, E. A. Theodorou, Constrained covariance steering based tube-mppi, in: 2022 American Control Conference (ACC), IEEE, 2022, pp. 4197–4202.

[57] J. Yin, P. Tsiotras, K. Berntorp, Chance-constrained information-theoretic stochastic model predictive control with safety shielding, in: 2024 IEEE 63rd Conference on Decision and Control (CDC), IEEE, 2024, pp. 653–658.

[58] I. S. Mohamed, M. Ali, L. Liu, Chance-constrained sampling-based mpc for collision avoidance in uncertain dynamic environments, IEEE Robotics and Automation Letters (2025).

[59] E. Trevisan, K. A. Mustafa, G. Notten, X. Wang, J. Alonso-Mora, Dynamic risk-aware mppi for mobile robots in crowds via efficient monte carlo approximations, arXiv preprint arXiv:2506.21205 (2025).

[60] J. Yin, Z. Zhang, P. Tsiotras, Risk-aware model predictive path integral control using conditional value-at-risk, in: 2023 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2023, pp. 7937–7943.

[61] H. Parwana, M. Black, B. Hoxha, H. Okamoto, G. Fainekos, D. Prokhorov, D. Panagou, Risk-aware mppi for stochastic hybrid systems, in: 2025 American Control Conference (ACC), IEEE, 2025, pp. 4369–4376.

[62] J. Yin, C. Dawson, C. Fan, P. Tsiotras, Shield model predictive path integral: A computationally efficient robust mpc method using control barrier functions, IEEE Robotics and Automation Letters 8 (11) (2023) 7106–7113.

[63] J. Yin, O. So, E. Yang Yu, C. Fan, P. Tsiotras, Safe beyond the horizon: Efficient sampling-based mpc with neural control barrier functions, in: Proceedings of Robotics: Science and Systems, 2025.

[64] Q. Zhao, H. Liu, C. Xia, J. Tan, B. Liang, Make it safer and stabler: Model predictive path integral control strategy using lyapunov function, in: 2025 11th International Conference on Control, Automation and Robotics (ICCAR), IEEE, 2025, pp. 232–239.

[65] P. Rabiee, J. B. Hoagg, Guaranteed-safe mppi through composite control barrier functions for efficient sampling in multi-constrained robotic systems, arXiv preprint arXiv:2410.02154 (2024).

[66] C. Tao, H.-J. Yoon, H. Kim, N. Hovakimyan, P. Voulgaris, Path integral methods with stochastic control barrier functions, in: 2022 IEEE 61st Conference on Decision and Control (CDC), IEEE, 2022, pp. 1654–1659.

[67] J. Borquez, L. Raus, Y. U. Ciftci, S. Bansal, Dual-guard mppi: Safe and performant optimal control by combining sampling-based mpc and hamilton-jacobi reachability, IEEE Robotics and Automation Letters (2025).

[68] F. Wang, H. Jiang, C. Tao, W. Wan, Y. Cheng, Dbas-log-mppi: Efficient and safe trajectory optimization via barrier states, arXiv preprint arXiv:2504.06437 (2025).

[69] T. Miura, N. Akai, K. Honda, S. Hara, Spline-interpolated model predictive path integral control with stein variational inference for reactive navigation, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2024, pp. 13171–13177.

[70] L. Park, K. Jang, S. Kim, Csc-mppi: A novel constrained mppi framework with dbscan for reliable obstacle avoidance, arXiv preprint arXiv:2506.16386 (2025).

[71] E. M. Andrejev, A. Manoharan, K.-E. Unt, A. K. Singh, π-mppi: A projection-based model predictive path integral scheme for smooth optimal control of fixed-wing aerial vehicles, IEEE Robotics and Automation Letters (2025).

[72] M.-G. Kim, M. Jung, J. Hong, K.-K. K. Kim, Mppi-ipddp: A hybrid method of collision-free smooth trajectory generation for autonomous robots, IEEE Transactions on Industrial Informatics (2025).

[73] J. Sacks, R. Rana, K. Huang, A. Spitzer, G. Shi, B. Boots, Deep model predictive optimization, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2024, pp. 16945–16953.

[74] J. Knaup, J. D'sa, B. Chalaki, T. Naes, H. N. Mahjoub, E. Moradi-Pari, P. Tsiotras, Active learning with dual model predictive path-integral control for interaction-aware autonomous highway on-ramp merging, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2024, pp. 14191–14197.

[75] D.-H. Nam, H.-G. Kim, Mppi-based integrated task assignment, path planning, and control for unmanned ground vehicles, International Journal of Control, Automation and Systems 22 (12) (2024) 3641–3652.

[76] J. Li, A. Wang, Y. Lyu, Z. Ding, Global path guided model predictive path integral control: Applications to gpu-parallelizable robot simulation systems, Computers and Electrical Engineering 120 (2024) 109645.

[77] I. S. Mohamed, M. Ali, L. Liu, Gp-guided mppi for efficient navigation in complex unknown cluttered environments, in: 2023 IEEE/RSJ international conference on intelligent robots and systems (IROS), IEEE, 2023, pp. 7463–7470.

[78] X. Zhang, J. Lu, Y. Hui, H. Shen, L. Xu, B. Tian, Rapa-planner: Robust and efficient motion planning for quadrotors based on parallel ra-mppi, IEEE Transactions on Industrial Electronics (2024).

[79] W. Jansma, E. Trevisan, A. Serra-Gómez, J. Alonso-Mora, Interaction-aware sampling-based mpc with learned local goal predictions, in: 2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS), IEEE, 2023, pp. 15–21.

[80] Y. Zhang, C. Pezzato, E. Trevisan, C. Salmi, C. H. Corbato, J. Alonso-Mora, Multi-modal mppi and active inference for reactive task and motion planning, IEEE Robotics and Automation Letters (2024).

[81] V. Vasilopoulos, S. Garg, P. Piacenza, J. Huh, V. Isler, Ramp: Hierarchical reactive motion planning for manipulation tasks using implicit signed distance functions, in: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2023, pp. 10551–10558.

[82] J. Pravitra, K. A. Ackerman, C. Cao, N. Hovakimyan, E. A. Theodorou, L1-adaptive mppi architecture for robust and agile control of multirotors, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2020, pp. 7661–7666.

[83] H. Homburger, S. Wirtensohn, J. Reuter, Mppi control of a self-balancing vehicle employing subordinated control loops, in: 2023 European Control Conference (ECC), IEEE, 2023, pp. 1–6.

[84] L. Zhou, Z. Li, Y. Li, S. Bai, Parallel mppi with gradient-velocity modulated sdf cost for high-performance real-time dynamic obstacle avoidance by robot manipulators, IEEE Transactions on Robotics (2025).

[85] M. Raisi, A. Noohian, S. Fallah, A fault-tolerant and robust controller using model predictive path integral control for free-flying space robots, Frontiers in Robotics and AI 9 (2022) 1027918.

[86] L. Jung, A. Estornell, M. Everett, Contingency constrained planning with mppi within mppi, in: 7th Annual Learning for Dynamics\& Control Conference, PMLR, 2025, pp. 869–880.

[87] I. Abraham, A. Handa, N. Ratliff, K. Lowrey, T. D. Murphey, D. Fox, Model-based generalization under parameter uncertainty using path integral control, IEEE Robotics and Automation Letters 5 (2) (2020) 2864–2871.

[88] F. F. Nyboe, A. Afifi, P. R. Giordano, E. Ebeid, A. Franchi, Embedded robust model predictive path integral control using sensitivity tubes and gpu acceleration, in: ICRA 2025-IEEE International Conference on Robotics & Automation, 2025.

[89] L. Cong, M. Grner, P. Ruppel, H. Liang, N. Hendrich, J. Zhang, Self-adapting recurrent models for object pushing from learning in simulation, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2020, pp. 5304–5310.

[90] E. Arruda, M. J. Mathew, M. Kopicki, M. Mistry, M. Azad, J. L. Wyatt, Uncertainty averse pushing with model predictive path integral control, in: 2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids), IEEE, 2017, pp. 497–502.

[91] D. Kalaria, H. Xue, W. Xiao, T. Tao, G. Shi, J. M. Dolan, Agile mobility with rapid online adaptation

via meta-learning and uncertainty-aware mppi, in: 2025 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2025, pp. 16626–16632.

[92] Z. Wang, G. Williams, E. A. Theodorou, Information theoretic model predictive control on jump diffusion processes, in: 2019 American Control Conference (ACC), IEEE, 2019, pp. 1663–1670.

[93] I. S. Mohamed, J. Xu, G. S. Sukhatme, L. Liu, Towards efficient mppi trajectory generation with unscented guidance: U-mppi control strategy, IEEE Transactions on Robotics (2025).

[94] J. Higgins, N. Mohammad, N. Bezzo, A model predictive path integral method for fast, proactive, and uncertainty-aware uav planning in cluttered environments, in: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2023, pp. 830–837.

[95] S. Patrick, E. Bakolas, Path integral control with rollout clustering and dynamic obstacles, in: 2024 American Control Conference (ACC), IEEE, 2024, pp. 809–814.

[96] G. Williams, B. Goldfain, P. Drews, K. Saigol, J. M. Rehg, E. A. Theodorou, Robust sampling based model predictive control with sparse objective information., in: Robotics: Science and Systems, Vol. 14, 2018, p. 2018.

[97] M. S. Gandhi, B. Vlahov, J. Gibson, G. Williams, E. A. Theodorou, Robust model predictive path integral control: Analysis and performance guarantees, IEEE Robotics and Automation Letters 6 (2) (2021) 1423–1430.

[98] C. Liang, W. Wang, Z. Liu, C. Lai, B. Zhou, Learning to guide: Guidance law based on deep meta-learning and model predictive path integral control, IEEE Access 7 (2019) 47353–47365.

[99] S. Folk, J. Melton, B. W. Margolis, M. Yim, V. Kumar, Towards safe and energy-efficient real-time motion planning in windy urban environments, in: 2025 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2025, pp. 2787–2793.

[100] B. Vlahov, J. Gibson, M. Gandhi, E. A. Theodorou, Mppi-generic: A cuda library for stochastic optimization, 2024, URL https://arxiv.org/abs/2409.07563.

[101] J. Alvarez-Padilla, J. Z. Zhang, S. Kwok, J. M. Dolan, Z. Manchester, Real-time whole-body control of legged robots with model-predictive path integral control, in: 2025 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2025, pp. 14721–14727.

[102] C. Pezzato, C. Salmi, E. Trevisan, J. A. Mora, C. H. Corbato, Sampling-based mpc using a gpu-parallelizable physics simulator as dynamic model: an open source implementation with isaacgym, in: Embracing Contacts-Workshop at ICRA 2023, 2023.

[103] C. Pezzato, C. Salmi, E. Trevisan, M. Spahn, J. Alonso-Mora, C. H. Corbato, Sampling-based model predictive control leveraging parallelizable physics simulations, IEEE Robotics and Automation Letters (2025).