Annual Reviews in Control 57 (2024) 100931

Contents lists available at ScienceDirect

Annual Reviews in Control

journal homepage: www.elsevier.com/locate/arcontrol

Review article

Recent advances in path integral control for trajectory optimization: An overview in theoretical and algorithmic perspectives

Muhammad Kazim, JunGee Hong, Min-Gyeom Kim, Kwang-Ki K. Kim [*,1]

Department of Electrical and Computer Engineering at Inha University, Incheon 22212, Republic of Korea

ARTICLE INFO

Keywords:
Stochastic optimal control
Trajectory optimization
Hamilton–Jacobi–Bellman equation
Feynman–Kac formula
Path integral
Variational inference
KL divergence
Importance sampling
Model predictive path integral control
Policy search
Policy improvement with path integrals
Planning on manifolds

ABSTRACT

This paper presents a tutorial overview of path integral (PI) approaches for stochastic optimal control and trajectory optimization. We concisely summarize the theoretical development of path integral control to compute a solution for stochastic optimal control and provide algorithmic descriptions of the cross-entropy (CE) method, an open-loop controller using the receding horizon scheme known as the model predictive path integral (MPPI), and a parameterized state feedback controller based on the path integral control theory. We discuss policy search methods based on path integral control, efficient and stable sampling strategies, extensions to multi-agent decision-making, and MPPI for the trajectory optimization on manifolds. For tutorial demonstrations, some PI-based controllers are implemented in Python, MATLAB and ROS2/Gazebo simulations for trajectory optimization. The simulation frameworks and source codes are publicly available at the github page.

1. Introduction

Trajectory optimization for motion or path planning (Betts, 1998; Rao, 2014; Von Stryk & Bulirsch, 1992) is a fundamental problem in autonomous systems (Choset, Lynch, Hutchinson, Kantor, & Burgard, 2005; Latombe, 2012; LaValle, 2006). Several requirements must be simultaneously considered for autonomous robot motion, path planning, navigation, and control. Examples include the specifications of mission objectives, examining the certifiable dynamical feasibility of a robot, ensuring collision avoidance, and considering the internal physical and communication constraints of autonomous robots.

In particular, generating an energy-efficient and collision-free safe trajectory is of the utmost importance during the process of autonomous vehicle driving (Claussmann, Revilloud, Gruyer, & Glaser, 2019; Paden, Čáp, Yong, Yershov, & Frazzoli, 2016; Teng et al., 2023), autonomous racing drone (Han et al., 2021; Hanover et al., 2023; Song, Steinweg, Kaufmann, & Scaramuzza, 2021), unmanned aerial vehicles (Lan, Lai, Lee, & Chen, 2021), electric vertical take-off and landing (eVTOL) urban air mobility (UAM) (Park, Kim, Suk, & Kim, 2023; Pradeep et al., 2020; Wang, Diepolder, Zhang, Söpper & Holzapfel, 2021), missile guidance (Kwon & Choi, 2020; Roh, Oh, Tahk, Kwon, & Kwon, 2020), space vehicle control, and satellite attitude trajectory optimization (Dearing, Hauser, Chen, Nicotra, & Petersen, 2022; Garcia & How, 2005; Gatherer & Manchester, 2019; Malyuta, Yu, Elango, & Açıkmeşe, 2021; Weiss, Leve, Baldwin, Forbes, & Kolmanovsky, 2014)

From an algorithmic perspective, the complexity of motion planning is NP-complete (Canny, 1988). Various computational methods have been proposed for motion planning, including sampling-based methods (Elbanhawi & Simic, 2014; Kingston, Moll, & Kavraki, 2018), nonlinear programming (NLP) (Betts, 2010; Kelly, 2017), sequential convex programming (SCP) (Bonalli, Cauligi, Bylard and Pavone, 2019; Howell, Jackson, & Manchester, 2019; Malyuta et al., 2022; Manyam, Casbeer, Weintraub, & Taylor, 2021), differential dynamic programming (DDP) (Cao, Cao, Yuan, & Xie, 2022; Chatzinikolaidis & Li, 2021; Chen, Zhan, & Tomizuka, 2019; Jacobson & Mayne, 1970; Kim & Kim, 2022a; Mayne, 1973; Pavlov, Shames, & Manzie, 2021; Xie, Liu, & Hauser, 2017), hybrid methods (Zhong, Tian, Hu, & Peng, 2020), and differential-flatness-based optimal control (Faessler, Franchi, & Scaramuzza, 2017; Sun, Romero, Foehn, Kaufmann, & Scaramuzza, 2022).

Optimization methods can explicitly perform safe and efficient trajectory generation for path and motion planning. The two most popular optimal path and motion planning methods for autonomous robots are

* Corresponding author.
E-mail address: kwangki.kim@inha.ac.kr (K.-K.K. Kim).

[1] This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), South Korea funded by the Ministry of Education (NRF-2022R1F1A1076260).

https://doi.org/10.1016/j.arcontrol.2023.100931
Received 11 September 2023; Received in revised form 1 December 2023; Accepted 30 December 2023
Available online 15 January 2024

gradient- and sampling-based methods for trajectory optimization. The former frequently assumes that the objective and constraint functions in a given planning problem are differentiable and can rapidly provide a locally optimal smooth trajectory (Domahidi & Jerez, 2014–2023). However, numerical and algorithmic computations of derivatives (gradient, Jacobian, Hessian, etc.) are not stable in the worst case; preconditioning and prescaling must be accompanied by and integrated into the solvers. In addition, integrating the outcomes of perception obtained from exteroceptive sensors such as LiDARs and cameras into collision-avoidance trajectory optimization requires additional computational effort in dynamic and unstructured environments.

Sampling-based methods for trajectory generation do not require function differentiability. Therefore, they are more constructive than the former gradient-based optimization methods for modeling obstacles without considering their shapes in the constrained optimization for collision-free path planning (Gammell & Strub, 2020; Kingston et al., 2018). In addition, sampling-based methods naturally perform explorations, thereby avoiding the local optima. However, derivative-free-sampling-based methods generally produce coarse trajectories with zigzagging and jerking movements. For example, rapidly exploring random trees (RRT) and probabilistic roadmap (PRM) methods generate coarse trajectories (Elbanhawi & Simic, 2014; Janson, Ichter, & Pavone, 2018; Kuffner & LaValle, 2000; LaValle, 2006). To mitigate the drawbacks of gradient- and sampling-based methods while maintaining their advantages, a hybrid method that combines them can be considered, as proposed in Campos-Macías, Gómez-Gutiérrez, Aldana-López, de la Guardia, and Parra-Vilchis (2017), Kiani et al. (2021), Ravankar, Ravankar, Emaru, and Kobayashi (2020) and Yu, Si, Li, Wang, and Song (2022).

Several open-source off-the-shelf libraries are available for implementing motion planning and trajectory optimization, which include Open Motion Planning Library (OMPL) (Sucan, Moll, & Kavraki, 2012), Stochastic Trajectory Optimization for Motion Planning (STOMP) (Kalakrishnan, Chitta, Theodorou, Pastor, & Schaal, 2011), Search-Based Planning Library (SBPL) (Likhachev, 2010), And Covariant Hamiltonian Optimization for Motion Planning (CHOMP) (Zucker et al., 2013).

The path integral (PI) for stochastic optimal control, which was first presented in Kappen (2005), is another promising approach for sampling-based real-time optimal trajectory generation. In the path integral framework, the stochastic optimal control associated with trajectory optimization is transformed into a problem of evaluating a stochastic integral, for which Monte Carlo importance sampling methods are applied to approximate the integral. It is also closely related to the cross-entropy method (Rubinstein & Kroese, 2004) for stochastic optimization and model-based reinforcement learning (Gómez, Kappen, Peters, & Neumann, 2014) for decision making. The use of path integral control has recently become popular with advances in high-computational-capability embedded processing units (Williams, Aldrich, & Theodorou, 2017) and efficient Monte Carlo simulation techniques (Rubinstein & Kroese, 2016; Thijssen & Kappen, 2018).

There are several variations of the PI control framework. The most widely used method in robotics and control applications is the model predictive path integral (MPPI) control, which provides a derivative-free sampling-based framework to solve finite-horizon constrained optimal control problems using predictive model-based random roll-outs in path-integral approximations (Williams, 2019; Williams et al., 2017; Williams, Drews, Goldfain, Rehg, & Theodorou, 2016, 2018). However, despite its popularity, the performance and robustness of the MPPI are degraded in the presence of uncertainties in the simulated predictive model, similar to any other model-based optimal control method. To take the plant-model mismatches of simulated roll-outs into account (Abraham et al., 2020; Pan, Theodorou, & Kontitsis, 2015), adaptive MPPI (Pravitra, Ackerman, Cao, Hovakimyan, & Theodorou, 2020), learning-based MPPI (Mohamed, Ali, & Liu, 2023; Okada, Aoshima, & Rigazio, 2017; Williams, Rombokas, & Daniel, 2015) and tube-based MPPI (Gandhi, Vlahov, Gibson, Williams, & Theodorou, 2021), uncertainty-averse MPPI (Arruda et al., 2017), fault-tolerant MPPI (Raisi, Noohian, & Fallah, 2022), safety-critical MPPI using control barrier function (CBF) (Tao, Yoon, Kim, Hovakimyan, & Voulgaris, 2022; Yin, Dawson, Fan & Tsiotras, 2023; Zeng, Zhang, & Sreenath, 2021), and covariance steering MPPI (Yin, Zhang, Theodorou, & Tsiotras, 2022) have been proposed. Risk-aware MPPI based on the conditional value-at-risk (CVaR) have also been investigated for motion planning with probabilistic estimation uncertainty in partially known environments (Barbosa, Lacerda, Duckworth, Tumova, & Hawes, 2021; Cai, Everett, Sharma, Osteen, & How, 2022; Wang, So, Lee and Theodorou, 2021; Yin, Zhang & Tsiotras, 2023).

The path integral (PI) control framework can also be combined with parametric and nonparametric policy search methods and improvements. For example, RRT is used to guide PI control for exploration (Tao, Kim, & Hovakimyan, 2023) and PI control is used to guide policy searches for open-loop control (Theodorou & Todorov, 2012), parameterized movement primitives (Ijspeert, Nakanishi, & Schaal, 2002; Theodorou, Buchli, & Schaal, 2010), and feedback control (Gómez et al., 2014; Levine & Koltun, 2013; Montgomery & Levine, 2016). To smoothen the sampled trajectories generated from the PI control, gradient-based optimization, such as DDP, is combined with MPPI (Kim & Kim, 2022b), regularized policy optimization based on the cross-entropy method is used (Thalmeier, Kappen, Totaro, & Gómez, 2020), and it is also suggested to smooth the resulting control sequences using a sliding window (Ruiz & Kappen, 2017; Särkkä, 2008) and a Savitzky–Golay filter (SGF) (Neve, Lefebvre, & Crevecoeur, 2022; Williams et al., 2018) and introducing an additional penalty term corresponding to the time derivative of the action (Kim, Park, Kwak, Bae, & Lee, 2022).

PI control is related to several other optimal control and reinforcement learning strategies. For example, variational methods of path integral optimal control are closely related to entropy-regularized optimal control (Lambert, Fishman, Fox, Boots, & Ramos, 2020; Lefebvre & Crevecoeur, 2022) and maximum entropy RL (MaxEnt RL) (Eysenbach & Levine, 2019; Theodorou et al., 2010). The path integral can be considered as a probabilistic inference for stochastic optimal control (Kappen, Gómez, & Opper, 2012; Watson, Abdulsamad, & Peters, 2020; Whittle, 1991) and reinforcement learning (Haarnoja, Tang, Abbeel, & Levine, 2017; Levine, 2018).

One of the most important technical issues in the practical application of path integral control is the sampling efficiency. Various importance sampling strategies have been suggested for rollouts in predictive optimal control. Several importance sampling (IS) algorithms exist (Martino, Elvira, Luengo, & Corander, 2015; Stich, Raj, & Jaggi, 2017) with different performance costs and benefits, as surveyed in Bugallo et al. (2017). Adaptive importance sampling (AIS) procedures are considered within the optimal control (Kappen & Ruiz, 2016; Thijssen & Kappen, 2018) and MPPI control (Asmar, Senanayake, Manuel, & Kochenderfer, 2023). In addition to the AIS algorithms, learned importance sampling (Carius, Ranftl, Farshidian, & Hutter, 2022), general Monte Carlo methods (Arouna, 2004; Rubinstein & Kroese, 2016), and cross-entropy methods (De Boer, Kroese, Mannor, & Rubinstein, 2005; Kobilarov, 2012; Zhang, Wang, Hartmann, Weber, & Schute, 2014) have been applied to PI-based stochastic optimal control.

Various case studies of PI-based optimal control have been published (Testouri, Elghazaly, & Frank, 2023): autonomous driving (Gandhi et al., 2021; Ha, Park, & Choi, 2019; Mohamed, Yin, & Liu, 2022; Williams et al., 2016), autonomous flying (Higgins, Mohammad, & Bezzo, 2023; Houghton, Oshin, Acheson, Theodorou, & Gregory, 2022; Mohamed, Allibert, & Martinet, 2020; Pravitra, Theodorou, & Johnson, 2021), space robotics (Raisi et al., 2022), autonomous underwater vehicles (Nicolay, Petillot, Marfeychuk, Wang, & Carlucho, 2023), and robotic manipulation planning (Hou, Wang, Zou, & Zhou, 2022; Yamamoto, Ariizumi, Hayakawa, & Matsuno, 2020).

Path integral strategies for optimal predictive control have also been adopted to visual servoing techniques (Costanzo, De Maria, Natale, & Russo, 2023; Mohamed, 2021; Mohamed, Allibert, & Martinet, 2021; Mohamed et al., 2022). Recently, the MPPI was integrated into Robot Operating Systems 2 (ROS 2) (Macenski, Moore, Lu, Merzlyakov, & Ferguson, 2023), an open-source production-grade robotics middleware framework (Macenski, Foote, Gerkey, Lalancette, & Woodall, 2022).

We expect that more applications of path integral control will emerge, particularly with a focus on trajectory optimization of motion planning for autonomous systems such as mobile robots, autonomous vehicles, drones, and service robots. In addition, it has been shown that path integral control can be easily extended to the cooperative control of multi-agent systems (Gómez, Thijssen, Symington, Hailes, & Kappen, 2016; Thijssen, 2016; Van Den Broek, Wiegerinck, & Kappen, 2008; Varnai & Dimarogonas, 2022; Wan, Gahlawat, Hovakimyan, Theodorou, & Voulgaris, 2021a).

There are still issues that must be addressed for scalable learning with safety guarantees in path integral control and its extended variations.

- Exploration–exploitation tradeoff is still not trivial,
- Comparisons of MPC-like open-loop control and parameterized state-feedback control should be further investigated as problem and task-specific policy parameterization itself is not trivial,
- Extensions to output-feedback and dual control have not yet been studied, and
- Extensions to cooperative and competitive multi-agent trajectory optimization with limited inter-agent measurements and information should be further investigated.

The remainder of this paper is organized as follows: Section 2 presents the overview of some path integral control methods. Section 3 reviews the theoretical background of path integral control and its variations. Section 4 describes the algorithmic implementation of several optimal control methods that employ a path-integral control framework. In Section 5, two MATLAB simulation case studies are presented to demonstrate the effectiveness of predictive path integral control. Section 6 presents the four ROS2/Gazebo simulation results of trajectory optimization for autonomous mobile robots, in which MPPI-based local trajectory optimization methods are demonstrated for indoor and outdoor robot navigation. In Section 7, extensions of path integral control to policy search and learning, improving sampling efficiency, multi-agent decision making, and trajectory optimization of manifolds are discussed. Section 8 concludes the paper and suggests directions for future research and development of path-integral control, especially for autonomous mobile robots.

2. Overview of path integral controllers

Path integral (PI) control methods stand at the forefront of contemporary research in stochastic optimal control and trajectory optimization. These techniques, primarily characterized by their robustness and versatility, have been developed to tackle complex control tasks under uncertainty. Central to these methods are Cross-Entropy Method (CEM), Model Predictive Path Integral (MPPI), and PI^2-CMA, each with distinct variations and algorithmic developments. In Fig. 1, we mentioned some hierarchical structure of path integral control methods. Below, we present a structured overview of these methods, outlining their types and key characteristics.

The Cross-Entropy Method is a probabilistic technique that iteratively refines control policies by minimizing a cost function and employing importance sampling. It has evolved significantly since its inception. Some of the CEM algorithms and there key characteristics are given in Table 1. MPPI is an open-loop controller using the receding horizon scheme. It has been pivotal for real-time control with various developments focusing on improving efficiency and reducing computational load. Some of the MPPI algorithms and there key characteristics

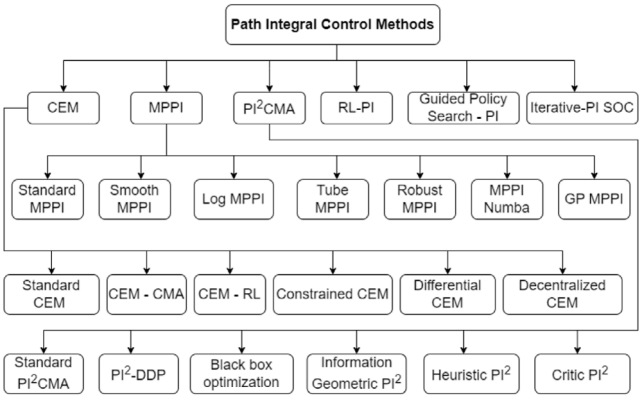

Fig. 1. Hierarchical classification of various path integral control methods.

are given in Table 2. PI^2-CMA represents an amalgamation of path integral control with Covariance Matrix Adaptation. It is especially suitable for problems where the cost landscape is highly non-convex or unknown. Some of the MPPI algorithms and there key characteristics are given in Table 3.

3. Path integral control: Theory

3.1. Stochastic optimal control

Controlled stochastic differential equation Consider a controlled stochastic differential equation of the following form:

$$dX_t = f(t, X_t, \pi(t, X_t))dt + g(t, X_t, \pi(t, X_t))dW_t, \tag{1}$$

where the initial condition is given by $X_0 = x_0 \in \mathbb{R}^n$, and W_t is a standard Brownian motion. The solution to the SDE (1) associated with the Markov policy $\pi : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^m$ is denoted by X_t^π.

Cost-to-go function For a given Markov policy π, the cost-to-go corresponding to policy π is defined as

$$G_t^\pi = \phi(X_T^\pi) + \int_t^T L(s, X_s^\pi, \pi(s, X_s^\pi))ds, \tag{2}$$

where $T > 0$ denotes the terminal time, $L : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ denotes the running cost, and $\phi : \mathbb{R}^n \to \mathbb{R}$ denotes the terminal cost.

Expected and optimal cost-to-go function The expected cost-to-go function is defined as

$$V^\pi(t, x) = \mathbb{E}[G_t^\pi | X_t^\pi = x] \tag{3}$$

where the expectation is considered with respect to the probability of the solution trajectory for the SDE (1) with an initial time and condition $(t, x) \in \mathbb{R} \times \mathbb{R}^n$. The goal of the stochastic optimal control is to determine the optimal policy.

$$\pi^* = \arg\min_\pi V^\pi(t, x) \tag{4}$$

The corresponding optimal cost-to-go function is defined as

$$V^*(t, x) = V^{\pi^*}(t, x) = \min_\pi V^\pi(t, x) \tag{5}$$

for each $(t, x) \in [0, T] \times \mathbb{R}^n$,

3.2. The Hamilton–Jacobi–Bellman equation

The Hamilton–Jacobi–Bellman equation for the optimal cost-to-go function is defined as follows (Fleming & Soner, 2006)

$$\min_\pi \left\{ L(t, x, \pi(t, x)) + \mathcal{T}^\pi V^*(t, x) \right\} = 0 \tag{6}$$

Table 1
Types of CEM algorithms and their description.

Type	Description	Ref.
Standard CEM	The standard CEM is a unified probabilistic approach for tackling combinatorial optimization, Monte-Carlo simulation, and machine learning challenges.	Rubinstein and Kroese (2004)
CEM-RL	CEM-RL merges CEM and TD3 to enhance DRL, optimizing both performance and sample efficiency.	Pourchot and Sigaud (2018)
Constrained CEM	Constrained CEM is a safety-focused reinforcement learning method that learns feasible policies while effectively tracking and satisfying constraints.	Wen and Topcu (2018)
Differentiable CEM	This introduces a gradient-based optimization into the CEM framework to improve the convergence rate.	Amos and Yarats (2020)
Decentralized CEM	Decentralized CEM improves Cross-Entropy Method efficiency by employing a distributed ensemble of CEM instances to mitigate local optima traps.	Zhang, Jin, Jagersand, Luo, and Schuurmans (2022)

Table 2
Types of MPPI algorithms and their description.

Type	Description	Ref.
Standard MPPI	MPPI leverages parallel optimization and generalized importance sampling for efficient navigation in nonlinear systems.	Williams et al. (2017)
Robust MPPI	It enhances off-road navigation with an augmented state space and tailored controllers, offering improved performance and robustness.	Gandhi et al. (2021)
Smooth MPPI	It enhances MPPI by incorporating input-lifting and a novel action cost to effectively reduce command chattering in nonlinear control tasks.	Kim et al. (2022)
Log MPPI	It enhances robotic navigation by using a normal log-normal mixture for efficient and feasible trajectory sampling in complex environments.	Mohamed et al. (2022)
Tube MPPI	Tube MPPI merges MPPI with Constrained Covariance Steering for robust, constraint-efficient trajectory optimization in stochastic systems.	Balci, Bakolas, Vlahov, and Theodorou (2022)
MPPI numba	This research develops a neural network-based, risk-aware method for off-road navigation, enhancing success and efficiency by analyzing empirical traction distributions.	Cai et al. (2022)
GP-MPPI	GP-MPPI combines MPPI with sparse Gaussian Processes for improved autonomous navigation in complex environments by learning model uncertainties and disturbances, and optimizing local paths.	Mohamed et al. (2023)

$(t,x) \in [0,T] \times \mathbb{R}^n$. where

$$\mathcal{T}^{\pi} V^*(t,x) = \lim_{h \to 0+} \left(\mathbb{E}[V^*(t+h, X^{\pi}_{t+h}) | X^{\pi}_t = x] - V^*(t,x) \right) \tag{7}$$

is a backward evolution operator defined on the functions of the class $C^1 \times C^2$. Additionally, the boundary condition is given by $V^*(T,x) = \phi(x)$.

3.3. Linearization of the HJB PDE

Control affine form and a quadratic cost As a special case of (1), we consider the controlled dynamics (diffusion process)

$$dX_t = f(t, X_t)dt + g(t, X_t)\left(\pi(t, X_t)dt + dW_t\right), \tag{8}$$

and the cost-to-go

$$G^{\pi}_t = \phi(X^{\pi}_T) + \int_t^T \pi(s, X^{\pi}_s)^{\top} dW_s + \int_t^T \left(q(s, X^{\pi}_s) + \frac{1}{2}\pi(s, X^{\pi}_s)^{\top} \pi(s, X^{\pi}_s) \right) ds \tag{9}$$

where X^{π}_s is the solution to the SDE (8) for $s \in [t,T]$ with the initial condition $X^{\pi}_t = x_t$. In this study, we assume $G^{\pi}_t > 0$ for all $t \in [0,T]$ and any (control) policy π.

Remark. Notice that

(1) G^{π}_t is not adaptive with respect to the Brownian motion as it depends on (X^{π}_{τ}) for $\tau > t$.

(2) The second term in (9) is a stochastic integral with respect to the Brownian motion and it vanishes when taking expectation. This term will play an essential role when applying a change of measure. □

The goal of stochastic optimal control for the dynamics (8) and the cost-to-go (9) is to determine an optimal policy that minimizes the expected cost-to-go with respect to the policy.

$$V^*(t,x) := \min_{\pi} \mathbb{E}[G^{\pi}_t | X^{\pi}_t = x], \tag{10}$$

$$\pi^*(t,x) := \arg\min_{\pi} V^{\pi}(t,x), \tag{11}$$

where the expectation $\mathbb{E}$ is considered with respect to the stochastic process $X^{\pi}_{t:T} \sim \mathcal{P}^{\pi}$ which is the (solution) path of SDE (8).

Table 3
Types of PI^2-CMA algorithms and their description.

Type	Description	Ref.
Standard PI^2-CMA	It is an algorithm that merges PI^2-CMA to optimize policies and auto-tune exploration noise in reinforcement learning.	Stulp and Sigaud (2012a)
Black-box optimization	It examines the convergence of RL and black-box optimization in PI, introducing the efficient PIBB algorithm.	Stulp and Sigaud (2012b)
PI^2-DDP	It is an advanced RL method that combines the gradient extraction principles of PI^2-CMA with the feedback mechanism of DDP.	Lefebvre and Crevecoeur (2019)
Information geometric PI^2	It reexamines PI^2, linking it with evolution strategies and information-geometric optimization to refine its cost minimization approach using natural gradient descent.	Varnai and Dimarogonas (2020)
Heuristic PI^2	It introduces PI2-CMA-KCCA, a RL algorithm that accelerates robot motor skill acquisition by using heuristic information from Kernel Canonical Correlation Analysis and CMA to guide Path Integral Policy Improvement.	Fu, Li, Teng, Luo, and Li (2020)
Critic PI^2	Critic PI^2 combine trajectory optimization and deep actor–critic learning in a model-based RL framework for improved efficiency in continuous control tasks.	Ba, Fan, Guo, and Hao (2021)

Theorem 1 (*Fleming & Rishel, 2012; Fleming & Soner, 2006; Oksendal, 2013*)**.** *The solution of the stochastic optimal control in (10) and (11) is given as*

$$V^*(t,x) = -\ln \mathbb{E}\left[e^{-G_t^\pi} | X_t^\pi = x\right], \tag{12}$$

and

$$\pi^*(t,x) = \pi(t,x) + \lim_{s\to t+} \frac{\mathbb{E}[(W_s - W_t)e^{-G_t^\pi} | X_t^\pi = x]}{\mathbb{E}[(s-t)e^{-G_t^\pi} | X_t^\pi = x]}, \tag{13}$$

where $\pi(t,x)$ denotes an arbitrary Markov policy. ■

Because the solution represented in Theorem 1 is defined in terms of a path integral for which the expectation $\mathbb{E}$ is taken with respect to the random process $X_{t:T}^\pi \sim \mathcal{P}^\pi$, that is, the (solution) path of the SDE (8), this class of stochastic optimal control with control-affine dynamics and quadratic control costs is called the path integral (PI) control.

Solution of the HJB equation For stochastic optimal control of the dynamics (8) and (9), the HJB equation can be rewritten as (Oksendal, 2013)

$$\min_u \left\{ q + \frac{1}{2}u^\top u + \mathcal{T}^u V^* \right\} = 0 \tag{14}$$

where

$$\mathcal{T}^u V^* = \partial_t V^* + (f + gu)^\top \partial_x V^* + \frac{1}{2}\mathrm{Tr}(gg^\top \partial_{xx} V^*) \tag{15}$$

with the boundary condition $V^*(T,x) = \phi(x)$. In addition, the optimal state-feedback controller is given by

$$u^*(t,x) = -g(t,x)^\top \partial_x V^*(t,x). \tag{16}$$

Here, Markov policy π is replaced by state-feedback control u without loss of generality. The value (i.e., the optimal expected cost-to-go) function V^* is defined as a solution to the second-order PDE (Fleming & Soner, 2006; Oksendal, 2013)

$$0 = q + \partial_t V^* - \frac{1}{2}(\partial_x V^*)^\top gg^\top \partial_x V^* + f^\top \partial_x V^* + \frac{1}{2}\mathrm{Tr}(gg^\top \partial_{xx} V^*). \tag{17}$$

Linearization via exponential transformation We define an exponential transformation as follows:

$$\psi(t,x) = \exp\left(-\frac{1}{\lambda}V^*(t,x)\right) \tag{18}$$

that also belongs to class $C^1 \times C^2$ provided $V^*(t,x)$ does. Applying the backward evolution operator of the *uncontrolled* process, that is, $u = 0$, to the function $\psi(t,x)$, we obtain

$$\mathcal{T}^0 \psi = \partial_t \psi + f^\top \partial_x \psi + \frac{1}{2}\mathrm{Tr}(gg^\top \partial_{xx}\psi) = \frac{1}{\lambda} q\psi \tag{19}$$

which is a linear PDE with the boundary condition $\psi(T,x) = \exp(-\phi(x)/\lambda)$. This linear PDE is known as the backward Chapman–Kolmogorov PDE (Oksendal, 2013).

3.4. The Feynman–Kac formula

The Feynman–Kac lemma (Oksendal, 2013) provides a solution to the linear PDE (19)

$$\psi(t,x) = \mathbb{E}\left[\exp\left(\left(-\frac{1}{\lambda}\int_t^T q(s, X_s^0)ds\right)\psi(T, x_T)\right)\right] \tag{20}$$

where $\psi(T, x_T) = \exp(-\phi(x)/\lambda)$. In other words,

$$\psi(t,x) = \mathbb{E}\left[\exp\left(-\frac{1}{\lambda}G_t^0\right) \middle| X_t^0 = x\right] \tag{21}$$

where the expectation $\mathbb{E}$ is taken with respect to the random process $X_{t:T}^0 \sim \mathcal{P}^0$; that is, the (solution) path of the uncontrolled version of the SDE (8)

$$dX_t^0 = f(t, X_t^0)dt + g(t, X_t^0)dW_t \tag{22}$$

and the uncontrolled cost-to-go is given by

$$G_t^0 = \phi(x_T^0) + \int_t^T q(s, X_s^0)ds \tag{23}$$

which is again a nonadaptive random process. From the definition of ψ, this gives us a path-integral form for the value function:

$$V^*(t,x) = -\lambda \ln \mathbb{E}\left[\exp\left(-\frac{1}{\lambda}G_t^0\right) \middle| X_t^0 = x\right] \tag{24}$$

$(t,x) \in [0,T] \times \mathbb{R}^n$.

3.5. Path integral for stochastic optimal control

Path integral control Although the Feynman–Kac formula presented in Section 3.4 provides a method to compute or approximate the value function (24), it is still not trivial to obtain an optimal Markov policy because the optimal controller in (16) is defined in terms of the gradient

of V^*, not by V^*. From (16) and Theorem 1, combined with the path integral control theory (Kappen, 2007, 2011; Theodorou, 2011; Thijssen, 2016; Thijssen & Kappen, 2015; Williams, 2019), we have

$$\begin{aligned} u^*(t,x) &= -g(t,x)^\top \partial_x V^*(t,x) \\ &= g(t,x)^\top \partial_x \ln \psi(t,x) \\ &= g(t,x)^\top \lim_{s\to t+} \frac{\mathbb{E}_{\mathcal{P}^0}[\exp\left(-\frac{1}{\lambda}G_t^0\right)\int_t^s g(\tau,X_\tau)dW_\tau]}{(s-t)\mathbb{E}_{\mathcal{P}^0}[\exp\left(-\frac{1}{\lambda}G_t^0\right)]} \end{aligned} \tag{25}$$

where the initial condition is $X_t^0 = x$. This is equivalent to (13) in Theorem 1.

Information theoretic stochastic optimal control Regularized cost-to-go function

$$S_t(\mathcal{P}^\pi) = G_t^0 + \lambda \ln \frac{d\mathcal{P}^\pi}{d\mathcal{P}^0} \tag{26}$$

where G_t^0 is the state-dependent cost given in (23) and $\frac{d\mathcal{P}^\pi}{d\mathcal{P}^0}$ is the Radon–Nikodym derivative[2] for the probability measures $\mathcal{P}^\pi$ and $\mathcal{P}^0$. Total expected cost function

$$\begin{aligned} \mathcal{V}_t(\mathcal{P}^\pi) &= \mathbb{E}_{\mathcal{P}^\pi}[S_t(\mathcal{P}^\pi)] \\ &= \mathbb{E}_{\mathcal{P}^\pi}[G_t^0] + \lambda D_{\mathrm{KL}}(\mathcal{P}^\pi \| \mathcal{P}^0) \end{aligned} \tag{27}$$

is known as the free energy of a stochastic control system (Fleming & McEneaney, 1995; Fleming & Soner, 2006; Theodorou, 2015; Theodorou & Todorov, 2012). There is an additional cost term for the KL divergence between $\mathcal{P}^\pi$ and $\mathcal{P}^0$ which we can interpret as a control cost. From Girsanov's theorem (Liptser & Shiriaev, 2001a, 2001b), we obtain the following expression for the Radon–Nikodym derivative corresponding to the trajectories of the control-affine SDE (8).

$$\frac{d\mathcal{P}^\pi}{d\mathcal{P}^0} = \exp\left(\int_t^T \frac{1}{2}\|u_s\|^2 ds + u_s^\top dW_s\right) \tag{28}$$

where $u_s = \pi(s, X_s^\pi)$ is the control input and the initial condition $X_t^\pi = X_t^0 = x_t$ with an initial time $t \in [0, T]$ can be arbitrary.

The goal of KL control is to determine a probability measure that minimizes the expected total cost

$$\mathcal{P}^* = \mathcal{P}^{\pi^*} = \arg\min_\pi \mathcal{V}_t(\mathcal{P}^\pi) = \arg\min_{\mathcal{P}\in\boldsymbol{\Delta}^\pi} \mathcal{V}_t(\mathcal{P}), \tag{29}$$

provided $\mathcal{V}_t^* = \inf_{\mathcal{P}\in\boldsymbol{\Delta}^\pi} \mathcal{V}_t(\mathcal{P})$ exists, where $\boldsymbol{\Delta}^\pi$ denotes the space of probability measures corresponding to a policy π.

Theorem 2 (*Thijssen, 2016; Thijssen & Kappen, 2015*). *The optimal regularized cost-to-go has zero variance and is the same as the expected total cost*

$$S_t(\mathcal{P}^*) = \mathcal{V}_t(\mathcal{P}^*) = -\lambda \ln \mathbb{E}_{\mathcal{P}^0}[\exp(-G_t^0/\lambda)]. \tag{30}$$

which is equivalent to (24) given in Section 3.4. ■

In addition, the Radon–Nikodym derivative is given by

$$\frac{d\mathcal{P}^*}{d\mathcal{P}^0} = \frac{\exp(-G_t^0/\lambda)}{\mathbb{E}_{\mathcal{P}^0}[\exp(-G_t^0/\lambda)]} \tag{31}$$

and combining (31) with the R-N derivative (28), we obtain

$$\begin{aligned} \frac{d\mathcal{P}^*}{d\mathcal{P}^\pi} &= \omega_t^\pi \\ &= \exp\left(-\int_t^T \frac{1}{2}\|u_s\|^2 ds - u_s^\top dW_s - \frac{1}{\lambda}G_t^0\right) \end{aligned} \tag{32}$$

where $u_s = \pi(s, X_s)$ is the control input following the policy $\pi : \mathbb{R}\times\mathbb{R}^n \to \mathbb{R}^m$, and ω_t^π is known as the importance weight (Kappen & Ruiz, 2016; Thijssen & Kappen, 2015) that is also a random process.

[2] This R-N derivative $\frac{d\mathcal{P}^\pi}{d\mathcal{P}^0}$ denotes the density of $\mathcal{P}^\pi$ relative to $\mathcal{P}^0$. We assume that $\mathcal{P}^\pi$ is absolutely continuous with respect to $\mathcal{P}^0$, denoted by $\mathcal{P}^\pi \ll \mathcal{P}^0$.

4. Path integral control: Algorithms

4.1. MC integration: Model-based rollout

Let a tuple $(\Omega, \mathcal{F}, \mathcal{Q})$ be a probability space with a random variable X and consider the function $\ell(\boldsymbol{X}) = \int_0^T L(\boldsymbol{X}_t)dt$ or $\ell(X) = L(\boldsymbol{X}_T)$. The main idea of path integral control is to compute the expectation

$$\rho = \mathbb{E}_{\mathcal{Q}}[\ell(\boldsymbol{X})] \tag{33}$$

using sampling-based methods, such as Monte Carlo simulations. In principle, function $\ell : \Omega \to \mathbb{R}$ can be any arbitrary real-valued function. A well-known drawback of Monte Carlo (MC) integration is its high variance.

Importance sampling The goal of importance sampling (Arouna, 2004; Rubinstein & Kroese, 2016) in MC techniques is to minimize the variance in the MC estimation of integration, $\mathbb{E}_{\mathcal{Q}}[\ell(\boldsymbol{X})] = \mathbb{E}_{\mathcal{P}}[\ell(\boldsymbol{X})\frac{d\mathcal{Q}}{d\mathcal{P}}]$. To reduce the variance, we want to find a probability measure $\mathcal{P}$ on $(\Omega, \mathcal{F})$ with which an unbiased MC estimate for ρ is given by

$$\hat{\rho}(\mathcal{P}) = \frac{1}{N_s}\sum_{i=1}^{N_s} \ell(\boldsymbol{X}_i)\frac{d\mathcal{Q}}{d\mathcal{P}}(\boldsymbol{X}_i), \tag{34}$$

where the ith sampled path $\boldsymbol{X}_i$ is generated from density $\mathcal{P}$, denoted by $\boldsymbol{X}_i \sim \mathcal{P}$, for $i = 1, \dots, N_s$.

Multiple importance sampling Multiple-based probability measures can also be used for the MC estimation.

$$\hat{\rho}(\{\mathcal{P}^j\}_{j=1}^{N_p}) = \frac{1}{N}\sum_{j=1}^{N_p}\sum_{i=1}^{N_s^j} \ell(\boldsymbol{X}_i^j)\frac{d\mathcal{Q}}{d\mathcal{P}^j}(\boldsymbol{X}_i^j)\gamma^j(\boldsymbol{X}_i^j) \tag{35}$$

where $\boldsymbol{X}_i^j \sim \mathcal{P}^j$ for $i = 1, \dots, N_s^j$ and $j = 1, \dots, N_p$. Here, $N = \sum_{j=1}^{N_p} N_s^j$ is the total number of sampled paths, and the reweighting function $\gamma^j : \Omega \to \mathbb{R}$ can be any function that satisfies the relation

$$\ell(\boldsymbol{X}) \neq 0 \quad \Rightarrow \quad \frac{1}{N}\sum_{j=1}^{N_p} N_s^j \gamma^j(\boldsymbol{X}) = 1 \tag{36}$$

which guarantees that the resulting MC estimation $\hat{\rho}$ is unbiased (Rubinstein & Kroese, 2016; Thijssen & Kappen, 2018). For example, the flat function $\gamma^j(\boldsymbol{X}) = 1$ for all $\boldsymbol{X}$ or the balance-heuristic function $\gamma^j(\boldsymbol{X}) = N/\sum_{k=1}^{N_p} N_s^j \frac{d\mathcal{P}^k}{d\mathcal{P}^j}(\boldsymbol{X})$ can be employed (Thijssen & Kappen, 2018).

4.2. Cross entropy method for PI: KL control

The well-known cross-entropy (CE) method (Rubinstein & Kroese, 2004, 2016), which was originally invented for derivative-free optimization, can also be applied to trajectory generation by computing the following information theory optimization:

$$\begin{aligned} \pi^* &= \arg\min_\pi D_{\mathrm{KL}}(\mathcal{P}^*\|\mathcal{P}^\pi) \\ &= \arg\min_\pi \mathbb{E}_{\mathcal{P}^*}\left[\ln\frac{d\mathcal{P}^*}{d\mathcal{P}^\pi}\right] \\ &= \arg\min_\pi \mathbb{E}_{\mathcal{P}^\pi}\left[\frac{d\mathcal{P}^*}{d\mathcal{P}^\pi}\ln\frac{d\mathcal{P}^*}{d\mathcal{P}^\pi}\right] \\ &= \arg\min_\pi \mathbb{E}_{\boldsymbol{X}\sim\mathcal{P}^\pi}\left[\omega^\pi(\boldsymbol{X})\ln\omega^\pi(\boldsymbol{X})\right] \\ &= \arg\min_\pi \mathbb{E}_{\boldsymbol{X}\sim\mathcal{P}^{\tilde{\pi}}}\left[\omega^{\tilde{\pi}}(\boldsymbol{X})\ln\frac{\omega^\pi(\boldsymbol{X})}{\omega^{\tilde{\pi}}(\boldsymbol{X})}\right] \\ &= \arg\min_\pi \mathbb{E}_{\boldsymbol{X}\sim\mathcal{P}^{\tilde{\pi}}}\left[\omega^{\tilde{\pi}}(\boldsymbol{X})\ln\omega^\pi(\boldsymbol{X})\right] \end{aligned} \tag{37}$$

where the importance weight is defined as:

$$\omega^\pi(\boldsymbol{X}) = \frac{d\mathcal{P}^*(\boldsymbol{X})}{d\mathcal{P}^\pi(\boldsymbol{X})} = \omega^{\tilde{\pi}}(\boldsymbol{X})\frac{d\mathcal{P}^{\tilde{\pi}}(\boldsymbol{X})}{d\mathcal{P}^\pi(\boldsymbol{X})} \tag{38}$$

Algorithm 1 CE_trajopt

Input: K: Number of samples
N: Decision horizon
π^0: Initial (trial) policy
while not converged **do**
  Sample trajectories $\{\boldsymbol{X}_1, \cdots, \boldsymbol{X}_K\}$ from $\mathcal{P}^{\pi^i}$.
  Determine the elite set threshold: $\gamma_i = J^{\pi^i}(\boldsymbol{X}_\kappa)$
    where κ denotes the index of the K_e best
    sampled-trajectory with $K_e < K$.
  Compute the elite set of sampled-trajectories:
    $\mathcal{E}_i = \left\{ \boldsymbol{X}_k | J^{\pi^i}(\boldsymbol{X}_k) \le \gamma_i \right\}$
  Update the policy:
    $\pi^{i+1} = \arg\min_{\pi} \frac{1}{|\mathcal{E}_i|} \sum_{\boldsymbol{X}_k \in \mathcal{E}_i} J^{\pi}(\boldsymbol{X}_k)$
  Check convergence
end while

where $\tilde{\pi}$ is the baseline Markov policy. In addition, rewriting the cost function in the fourth row of (37) as $J^{\pi}(\boldsymbol{X}) := \omega^{\pi}(\boldsymbol{X}) \ln \omega^{\pi}(\boldsymbol{X})$, we have the following expectation minimization:

$$\min_{\pi} \mathbb{E}_{\boldsymbol{X}\sim\mathcal{P}^{\pi}} \left[J^{\pi}(\boldsymbol{X}) \right] . \tag{39}$$

Alg. 1 summarizes the iterative procedures of CE for motion planning (Kobilarov, 2012) to solve the optimization problem (39) using a sampling-based method.

Remark. For expectation minimization in (39) and Alg. 1, it is common to use a parameterization of the control policy π or the resulting trajectory distribution $\mathcal{P}^{\pi}$ which can be rewritten as $\mathcal{P}^{\pi}(\boldsymbol{X}) = \mathcal{P}(\boldsymbol{X};\theta)$. This parameterization results in a finite-dimensional optimization. □

4.3. MPC-like open-loop controller: MPPI

By applying time discretization with arithmetic manipulations to (25), the path integral control becomes

$$u^*(t,x) = u(t,x) + \frac{\mathbb{E}_{\mathcal{Q}}[\exp\left(-\frac{1}{\lambda} G_t^{\pi}\right) \delta u]}{\mathbb{E}_{\mathcal{Q}}[\exp\left(-\frac{1}{\lambda} G_t^{\pi}\right)]}$$

where u is the nominal control input and δu is the deviation control input for exploration. Here, the expectation is considered with respect to the probability measure $\mathcal{Q}$ of a path corresponding to policy π.

For implementation, the expectation is approximated using MC importance sampling as follows:

$$u^*(t,x) \approx u(t,x) + \sum_{k=1}^{K} \frac{\exp\left(-\frac{1}{\lambda} G_t^{\pi_k}\right)}{\sum_{\kappa=1}^{K} \exp\left(-\frac{1}{\lambda} G_t^{\pi_\kappa}\right)} \delta u_k(t,x)$$

where K is the number of sampled paths, and $G_t^{\pi_k}$ is the cost-to-go corresponding to the simulated trajectory following policy $\pi_k(t,x) = u(t,x) + \delta u_k(t,x)$ corresponding to the perturbed control inputs for $k = 1, 2, \ldots, K$. This path-integral control based on forward simulations is known as the model-predictive path integral (MPPI) (Williams, 2019; Williams et al., 2017, 2016). By recursively applying MPPI, the control inputs can approach the optimal points. Alg. 2 summarizes the standard procedures for the MPPI.

4.4. Parameterized state feedback controller

Although MPC-like open-loop control methods are easy to implement, they exhibit certain limitations. First, it can be inefficient for high-dimensional control spaces because a longer horizon results in a higher dimension of the decision variables. Second, it does not design a control law (i.e., policy), but computes a sequence of control inputs over a finite horizon, which means that whenever a new state is encountered, the entire computation should be repeated. Although a warm start can help solve this problem, it remains limited. Third, the trade-off between exploitation and exploration is not trivial.

Algorithm 2 MPPI_control

Input: K: Number of samples
N: Decision horizon
$(u_0, u_1, \ldots, u_{N-1})$: Initial control sequence
while not terminated **do**
  Generate random control variations δu
  for $k = 1, \ldots, K$ **do**
    $x_0 = x_{\text{init}}$
    $t_0 = t_{\text{init}}$
    for $i = 0, \ldots, N-1$ **do**
      $f_i = f(t_i, x_i)$
      $g_i = g(t_i, x_i)$
      $\tilde{u}_{i,k} = u_i + \delta u_{i,k}$
      $x_{i+1} = x_i + (f_i + g_i \tilde{u}_{i,k}) \Delta t$
      $t_{i+1} = t_i + \Delta t$
    end for
    $G_{N,k} = \texttt{cost}(x_N)$
    for $i = N-1, \ldots, 0$ **do**
      $G_{i,k} = G_{i+1,k} + \texttt{cost}(x_i, \tilde{u}_{i,k})$
    end for
  end for
  for $i = 0, \ldots, N-1$ **do**
    $w_{i,k} = \frac{\exp(-G_{i,k}/\lambda)}{\sum_{\kappa=1}^{K} \exp(-G_{i,\kappa}/\lambda)}$
    $u_i \leftarrow u_i + \sum_{k=1}^{K} w_{i,k} \delta u_{i,k}$
  end for
  Send u_0 to actuators
  for $i = 0, \ldots, N-1$ **do**
    $u_i = u_{\min(i+1, N-1)}$
  end for
  Update $x_{\text{init}}, t_{\text{init}}$ by measurement
end while

As an alternative to open-loop controller design, a parameterized policy or control law can be iteratively updated or learned via model-based rollouts, from which the performance of a candidate policy of parameterization is evaluated, and the parameters are updated to improve the control performance. The main computation procedure is that from an estimate of the probability $P(\boldsymbol{X}|x_0)$ of the sampled trajectories, we want to determine a parameterized policy $\pi_t(u_t|x_t;\theta_t)$ for each time $t < T$ that can reproduce the trajectory distribution $P(\boldsymbol{X}|x_0)$, where $\theta_t \in \Theta$ denotes the parameter vector that defines a feedback policy π_t (Gómez et al., 2014; Thijssen & Kappen, 2015). In general, a feedback policy can be time varying, and if it is time invariant, then the time dependence can be removed; that is, $\theta = \theta_t$ for all times $t \in [0, T]$. In this review paper, we consider only deterministic feedback policies, but the main idea can be trivially extended to probabilistic feedback policies[3].

Linearly parameterized state feedback Consider

$$u(t,x) = h(t,x)^{\top} \theta \tag{40}$$

[3] An example of probabilistic feedback policy parameterization is a time-dependent Gaussian policy that is linear in the states, $\pi_t(u_t|x_t;\theta_t) \sim \mathcal{N}(u_t|k_t + K_t x_t, S_t)$, in which the parameter vector is $\theta_t = (k_t, K_t, S_t)$ and updated by a weighted linear regression and the weighted sample-covariance matrix (Gómez et al., 2014; Kupcsik, Deisenroth, Peters, & Neumann, 2013).

where $h : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^{n_p}$ is a user-defined feature of the state feedback control law. Using the model-based forward simulations, a control parameter update rule can be applied as follows:

$$\theta \leftarrow \theta + \sum_{k=1}^{K} w_k \delta\theta_k \tag{41}$$

where $w_k = \frac{\exp\left(-\frac{1}{\lambda} G_t^{\pi_k}\right)}{\sum_{\kappa=1}^{K} \exp\left(-\frac{1}{\lambda} G_t^{\pi_\kappa}\right)}$ is the weight for the kth sampled perturbation-parameter $\delta\theta_k$ of the linearly parameterized control law and $\pi_k(t, x) = h(x, t)^\top(\theta + \delta\theta_k)$ is the test or the exploration (search) policy.

Nonlinearly parameterized state feedback Consider

$$u(t, x) = \pi(t, x; \theta) \tag{42}$$

where the state-feedback control law is parameterized by the control parameter $\theta \in \mathbb{R}^{n_p}$. Using the model-based forward simulations, a control parameter update rule can be applied as follows:

$$\theta \leftarrow \theta + \sum_{k=1}^{K} \tilde{w}_k \delta\theta_k \tag{43}$$

where the weight is defined as follows:

$$\tilde{w}_k = w_k [\nabla_\theta \pi(t, x; \theta)]^\dagger [\nabla_\theta \pi(t, x; \theta + \delta\theta_k)] \tag{44}$$

with the weight $w_k = \frac{\exp\left(-\frac{1}{\lambda} G_t^{\pi_k}\right)}{\sum_{\kappa=1}^{K} \exp\left(-\frac{1}{\lambda} G_t^{\pi_\kappa}\right)}$ and the exploration policy $\pi_k = \pi(t, x; \theta + \delta\theta_k)$. Here, $[\cdot]^\dagger$ denotes the pseudoinverse.

Remark (*CE Method for Policy Improvement*)**.** Parameterized state feedback controls, such as (40) and (42), can also be updated using CE methods. For example, $\delta\theta_k \sim \mathcal{GP}(0, \Sigma)$ is samples, the cost of simulated trajectories is evaluated with control parameters $\theta_k = \theta + \delta\theta_k$ for $k = 1, \ldots, K$, the samples are sorted in ascending order according to the simulated costs, and the parameter is updated by weighted-averaging the sampled parameters $\delta\theta_k$ from the sorted elite set, $\theta \leftarrow \theta + \texttt{average}(\delta\theta_k)_{\texttt{elite}}$. In general, the covariance Σ can be also updated by empirical covariance of the sampled parameters $\delta\theta_k$ from the sorted elite set, $\Sigma \leftarrow \Sigma + \texttt{average}(\delta\theta_k \delta\theta_k^\top)_{\texttt{elite}}$. □

4.5. Policy improvement with path integrals

Policy improvement with path integrals (PI^2) is presented in Theodorou et al. (2010). The main idea of PI^2 is to iteratively update the policy parameters by averaging the sampled parameters in weights with the costs of the path integral corresponding to the simulated trajectories (Yamamoto et al., 2020). Alg. 3 shows the pseudocode for the PI^2 Covariance Matrix Adaptation (PI^2-CMA) proposed in Stulp and Sigaud (2012a) based on the CMA evolutionary strategy (CMAES) (Hansen, 2016; Hansen & Ostermeier, 2001). In Stulp and Sigaud (2012a), the PI^2-CMA was compared with CE methods and CMAES in terms of optimality, exploration capability, and convergence rate. Skipping the covariance adaptation step in Alg. 3 yields a vanilla PI^2. In Theodorou et al. (2010), it was also shown that policy improvement methods based on PI^2 would outperform existing gradient-based policy search methods such as REINFORCE and NAC.

Algorithm 3 PI^2-CMA

Input: K: Number of samples
N: Decision horizon
(θ, Σ): Initial hyper-parameter
while not terminated **do**
  Generate random variables $\theta_{i,k} \sim \mathcal{GP}(\theta, \Sigma)$
  Generate random initial conditions x_{init}
  for $k = 1, \ldots, K$ **do**
    $x_0 = x_{\text{init}}$
    $t_0 = t_{\text{init}}$
    for $i = 0, \ldots, N-1$ **do**
      $f_i = f(t_i, x_i)$
      $g_i = g(t_i, x_i)$
      $u_{i,k} = \pi(t_i, x_i; \theta_{i,k})$
      $x_{i+1} = x_i + (f_i + g_i u_{i,k})\Delta t$
      $t_{i+1} = t_i + \Delta t$
    end for
    $G_{N,k} = \texttt{cost}(x_N)$
    for $i = N-1, \ldots, 0$ **do**
      $G_{i,k} = G_{i+1,k} + \texttt{cost}(x_i, u_{i,k})$
    end for
  end for
  for $i = 0, \ldots, N-1$ **do**
    $w_{i,k} = \frac{\exp(-G_{i,k}/\lambda)}{\sum_{\kappa=1}^{K} \exp(-G_{i,\kappa}/\lambda)}$
    $\theta_i = \sum_{k=1}^{K} w_{i,k} \theta_{i,k}$
    $\Sigma_i = \sum_{k=1}^{K} w_{i,k} (\theta_{i,k} - \theta)(\theta_{i,k} - \theta)^\top$
  end for
  $\theta \leftarrow \sum_{i=0}^{N-1} \frac{N-i}{\sum_{j=0}^{N-1}(N-j)} \theta_i$
  $\Sigma \leftarrow \sum_{i=0}^{N-1} \frac{N-i}{\sum_{j=0}^{N-1}(N-j)} \Sigma_i$
end while

5. Python and MATLAB simulation results

This section presents the simulation results of two different trajectory optimization problems: 1D cart–pole trajectory optimization and bicycle-like mobile robot trajectory tracking. The first case study demonstrates the simulation results for the cart–pole system, followed by the second numerical experiment showing the local trajectory tracking of a bicycle-like mobile robot with collision avoidance of dynamic obstacles.[4] The details of the simulation setups and numerical optimal control problems are not presented here because of limited space, but they are available in the accompanying github page.[5]

5.1. Cart–pole trajectory optimization

In this section, we consider trajectory optimization for set-point tracking in a one-dimensional (1D) cart–pole system. This system comprises an inverted pendulum mounted on a cart capable of moving along a unidirectional horizontal track. The primary goal is to achieve a swing-up motion of the pole and subsequently maintain its stability in an upright position through horizontal movements of the cart. We considered a quadratic form for the cost function associated with each state variable. To address this challenge of stabilizing a cart–pole system, we implemented several path integral methods including CEM, MPPI, and PI^2-CMA delineated in Algs. 1, 2, and 3, and investigated their performances with comparison to nonlinear model predictive control (NMPC).

Simulations are performed in Python environments. Fig. 2 illustrates the controlled trajectories of the cart–pole system under various control strategies. Fig. 3 depicts the force inputs derived from the application of each control method. These results demonstrate the effectiveness of the path integral control methods in achieving the desired control objectives for the cart–pole system. The comparative

[4] Videos of simulation results are available at
- https://youtu.be/zxUN0y23qio.
- https://youtu.be/LrYvrgju_o8.

[5] https://github.com/iASL/pic-review.git.

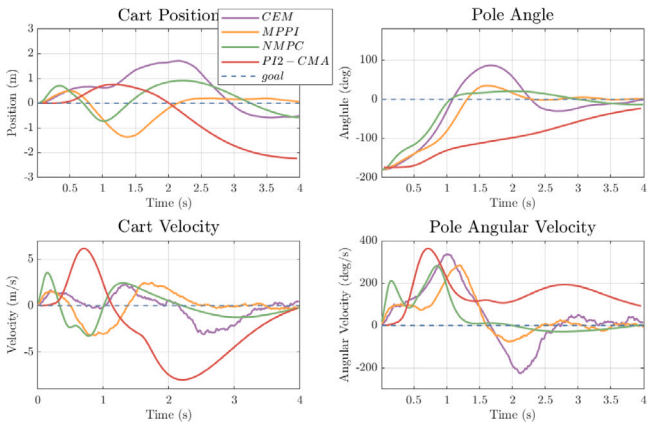

Fig. 2. Controlled trajectories of the 1D cart–pole using different methods of path integral control (CEM, MPPI,PI^2-CMA) in comparison with nonlinear model predictive control (NMPC).

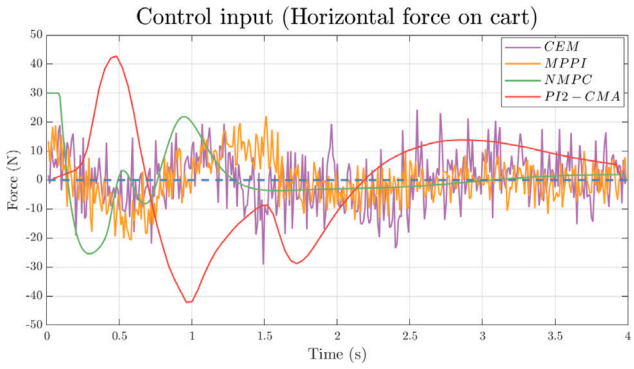

Fig. 3. Control inputs resulting from different methods of path integral control (CEM, MPPI,PI^2-CMA) and NMPC.

analysis of our results reveals distinct characteristics and performance efficiencies among the different path integral control methods. MPPI and CEM methods achieved better performance of stabilizing the pole as compared to NMPC and PI^2-CMA. Upon fine-tuning optimization parameters, MPPI achieved faster convergence in pole stabilization than CEM, given the same prediction horizon and sample size. Such enhanced performance of MPPI is further corroborated by its superior cost-to-go metrics, as shown in Fig. 4.

This comparison of four different PI-based control strategies stabilizing a cart–pole system reveals notable characteristics. CEM and MPPI result in smooth trajectories of the cart–pole position and lower accumulated cost-to-go, indicating better energy efficiency. In contrast, PI^2-CMA and NMPC result in more aggressive control input sequences with faster responses and larger overshoots in velocities, which might increase energy use and hasten actuator degradation. However, CEM and MPPI show higher rate of input changes, which would not be desirable for real hardware implementations. Of course, this jerking behavior of control actions can be relaxed by putting rate or ramp constraints in control inputs. The appropriate choice among these PI-based control strategies would heavily depend on the specific demands of the application, balancing multiple objectives such as efficiency, responsiveness, and operational durability.

5.2. Path planning for bicycle-like mobile robot

The robot system we consider in this section is a mobile robot navigating in an environment with obstacles, aiming to follow a predefined path while avoiding collisions and staying inside a track. To

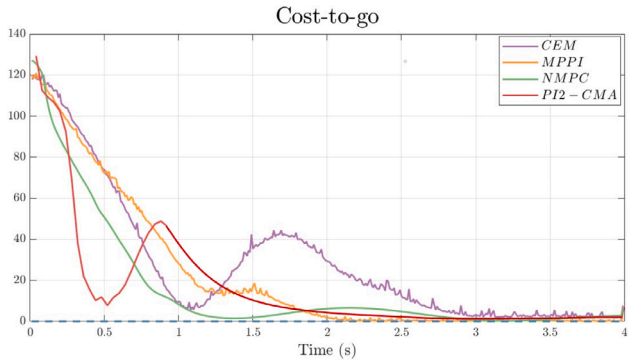

Fig. 4. Cost-to-go associated with different methods of path integral control (CEM, MPPI,PI^2-CMA) and NMPC.

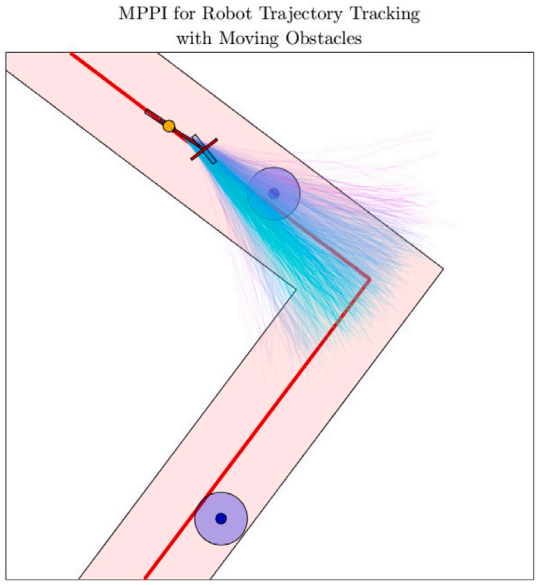

Fig. 5. A capture of simulating local path planning and tracking with obstacle avoidance using an MPPI controller. An associated video of simulations is available at https://youtu.be/LrYvrgju_o8.

evaluate the effectiveness of an MPPI controller for real-time trajectory generation with obstacle avoidance, we conducted a series of MATLAB simulations where two moving obstacles are considered in the 2D coordinates.

We implemented the MPPI controller described in Alg. 2 for bicycle-like mobile robot navigation over a track. Fig. 5 shows the robot's trajectory tracking while avoiding moving obstacles. Fig. 6 shows the forward and angular velocities of the robot required to follow the desired path by avoiding obstacles, which assesses the effectiveness of the controller in avoiding obstacles. The results demonstrated that the MPPI controller achieved successful trajectory generation and tracking, while effectively avoiding dynamic obstacles. Throughout the simulation of mobile robot path planning, MPPI controller showed robust obstacle avoidance capabilities, successfully navigating around obstacles, and minimizing the distance with its reference.

6. MPPI-based autonomous mobile robot navigation in ROS2/Gazebo environments

In this section, we consider two ROS2/Gazebo simulations of MPPI-based autonomous mobile robot navigation in a cafeteria environment and in a maze. We implement various path integral algorithms including MPPI, Smooth MPPI, and Log MPPI to enhance navigation and

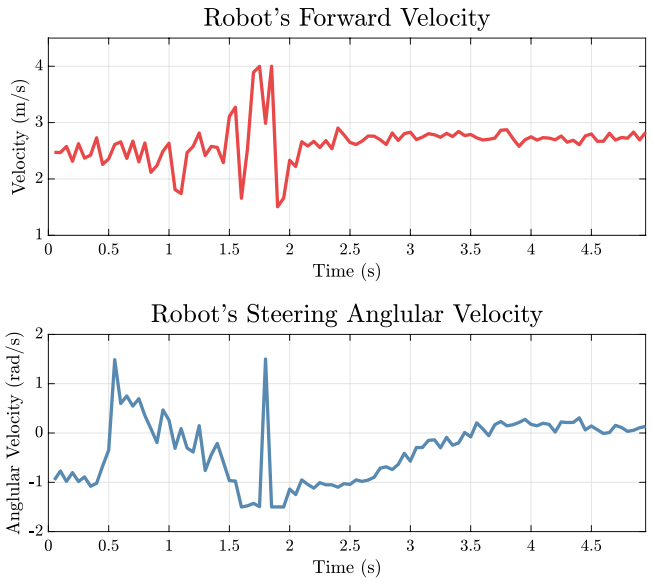

Fig. 6. Forward and angular velocity trajectories resulting from a MPPI-based controller steering a bicycle-like mobile robot.

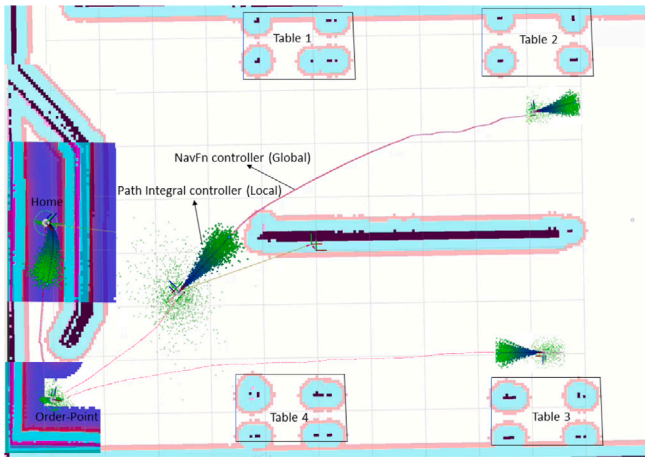

Fig. 7. ROS2/Gazebo simulation navigating an autonomous mobile robot in a cafeteria environment for which MPPI, Smooth MPPI, and Log MPPI are applied as local controllers. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

control in these complex scenarios. The SLAM toolbox is employed for mapping while Navigation2 (NAV2) is used for navigation within the ROS2 Gazebo simulation environment (Macenski et al., 2022, 2023). These path integral control approaches are attractive because they are derivative-free and can be parallelized efficiently on both CPU and GPU machines. The simulations are conducted using a computing system equipped with Intel Core i7 CPU and NVIDIA GeForce RTX 3070 GPU. These simulations are executed within the Ubuntu 22.04 operating system, leveraging the ROS2 Humble simulation platform for comprehensive analysis. The following subsections provide the details of the simulation setups and results of different MPPI-based path planning algorithms in two different scenarios. The simulation frameworks and source codes used in this study are publicly available.[6]

6.1. Autonomous robot navigation in a cafeteria environment

In this subsection, we present the simulation results of the autonomous robot navigation using different MPPI-based path planning algorithms in a cafeteria environment. The primary objective of this ROS2/Gazebo simulation is to evaluate the performance of MPPI-based path planning algorithms for a task of navigating an indoor autonomous mobile robot serving foods at different tables while avoiding obstacles in a confined and dense indoor space.

6.1.1. Experiment setup

For the autonomous indoor robot navigation scenario, we utilize an autonomous mobile robot equipped with the necessary sensors such as a LiDAR and a camera for environmental perception. The cafeteria environment is designed using a Gazebo, accurately replicating the challenges of indoor navigation, including cluttered spaces, narrow passages, and obstacles along a path generated by a global planner, such as `NavFn`, as shown in Fig. 7. All three path integral controllers run in a receding-horizon fashion with 100-time steps of prediction horizon where each time-step corresponds to 0.1 s The number of control rollouts is 1024 and the number of sampled traction maps is 2000 at the rate of 30 Hz. It can also replan at 30 Hz while sampling new control actions and maps.

In the simulated cafeteria scenario demonstrating the navigational efficiency in a dynamic service environment, as shown in Fig. 7, the robot is tasked to navigate a series of waypoints representing a service cycle. Commencing from the home coordinates ($x = -5, y = 0.51, \theta = 0.01$), the robot is required to proceed to the order-taking waypoint ($x = -4.85, y = -3.0, \theta = 0.01$). Subsequently, it traverses to serve Table 2 at the coordinates ($x = 4.29, y = 2.64, \theta = 0.01$) and upon order collection, returned to the order-taking waypoint. The mission is completed when the robot serves Table 3 at the coordinates ($x = -0.6, y = -1.99, \theta = 0.01$) and returns to the order-taking location. The total navigational distance required for the service unit to traverse, beginning from its home location, involves a multi-point itinerary. Initially, it must travel to the order point, followed by a journey to Table 2. Afterwards, it returns to the order point, from where it proceeds to serve Table 3, and subsequently returns to the order-taking point. This entire route encompasses a distance of approximately 43.15 m.

6.1.2. Simulation results

The simulation results of autonomous robot navigation in a cafeteria environment are given in Fig. 7 and the accompanying video,[7] to illustrate and compare the effectiveness of the three path integral controllers, MPPI, Smooth MPPI, and Log MPPI. In the simulated hotel navigation task, the performance of an autonomous delivery robot is evaluated using the `NavFn` global planner, in conjunction with three local planners: MPPI, Smooth MPPI, and Log MPPI. Figs. 8 and 9 demonstrate the longitudinal position and orientation tracking of the robot pose over time.

In the Gazebo simulations captured in Fig. 7, all three path integral controllers demonstrate remarkable indoor navigation capabilities for the differential-driving mobile robot, Turtlebot3. All three MPPI-based mobile robot navigation result in smooth trajectory tracking while effectively avoiding obstacles and collisions in crowded café environments. Fig. 8 shows the performance of the robot's positional components plotted over time. Each trajectory—MPPI (red line), Smooth MPPI (dark blue line), and Log MPPI (yellow line)—is compared against the ground truth (dotted_light red, dotted dark blue, and dotted green lines respectively). Despite slight deviations, the path integral controllers maintained a closely matched orientation with the ground truth, signifying robust direction control. Fig. 9 shows the robot's orientation (θ) as guided by the `NavFn` global planner with the three local planners.

[6] https://github.com/iASL/pic-review.git.

[7] https://youtu.be/3VChYScJ7oA.

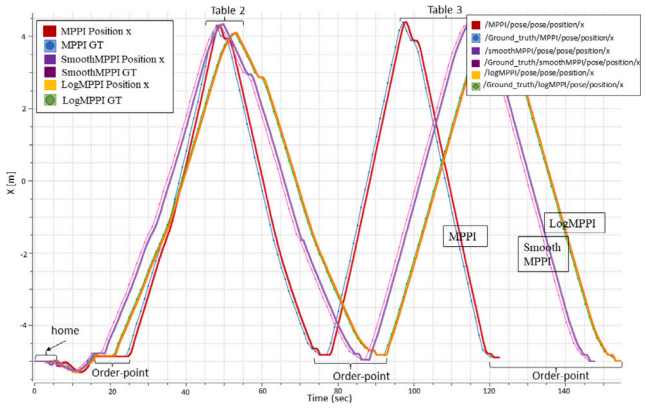

Fig. 8. Performance comparison of path integral controllers in terms of the longitudinal position (x). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

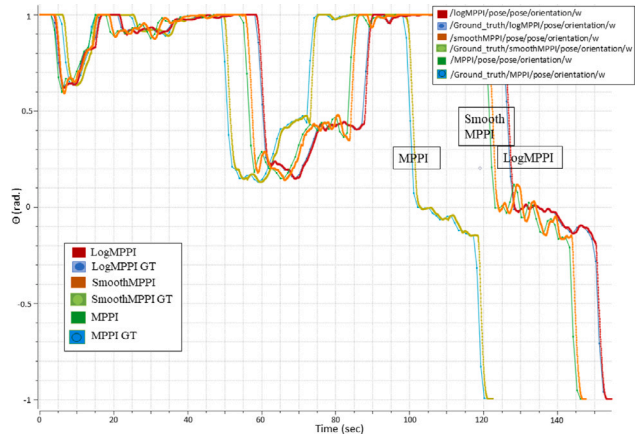

Fig. 9. Performance comparison of path integral controllers in terms of the orientation (θ). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Similar to Fig. 8, the tracking performance of the three algorithms is compared against the ground truth. In Figs. 8 and 9, the legends positioned in the top right corner display the Robot Operating System (ROS) topics pertaining to the robot's odometry and Gazebo ground truth data.

In the simulation study shown in Figs. 7, 8, and 9, all MPPI-based path integral algorithms demonstrated high precision in trajectory tracking. Notably, Smooth MPPI (represented by a dark blue line) maintained a trajectory closest to the ground truth, especially in the simulation's latter stages. While MPPI excelled in speed, taking less time, it exhibited higher positional and angular errors compared to Smooth MPPI and Log MPPI. Log MPPI required marginally more time than Smooth MPPI. However, in terms of both positional accuracy and angular orientation, Smooth MPPI outperformed both Log MPPI and MPPI. This highlights Smooth MPPI's superior balance of speed and precision in trajectory tracking in a cafeteria environment.

6.2. Autonomous robot navigation in maze-solving environment

In this subsection, we detail the simulation outcomes of utilizing the different path integral controllers for the navigation of the robot in a maze-solving environment. The primary goal of a maze-solving robot is to autonomously navigate through a labyrinth from a starting point to a designated endpoint in the shortest time possible. It must efficiently map the maze, identify the optimal path, and adjust to obstacles using its onboard sensors and algorithms.

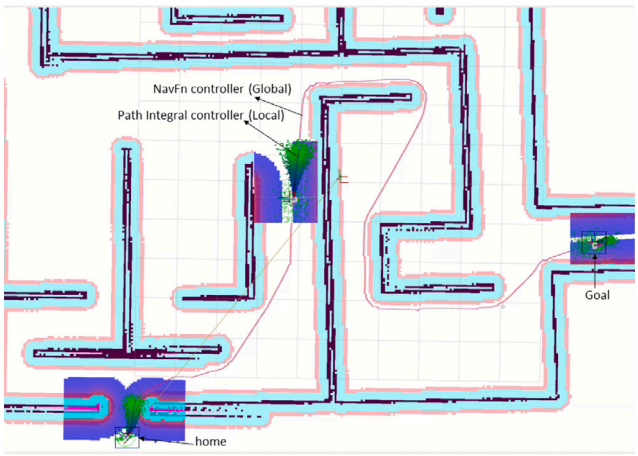

Fig. 10. ROS2/Gazebo simulation navigating an autonomous mobile robot in a maze-solving environment for which MPPI, Smooth MPPI, and Log MPPI are applied as local controllers.

6.2.1. Experimental setup

In our experiment, a simulated maze-solving robot was deployed in a ROS2/Gazebo environment, equipped with LiDAR and IMU sensors for navigation. The robot utilized the `NavFn` global planner for overall pathfinding and local controllers including MPPI, Smooth MPPI, and Log MPPI for fine-tuned maneuvering. Performance was assessed based on the robot's ability to discover the most efficient path to the maze center, measuring metrics such as completion time and path optimality. The path integral controllers ran in a receding-horizon fashion with 100-time steps; each step was 0.1 s. The number of control rollouts was 1024, and the number of sampled traction maps was 2000 at a rate of 30 Hz. It could also replan at 30 Hz while sampling new control actions and maps. The computational overhead of these algorithms remained reasonable, thereby ensuring real-time feasibility for practical applications.

In the simulated cafeteria scenario, as shown in Fig. 10, the autonomous robot was programmed to navigate from an initial location at coordinates ($x = -5.18, y = -6.58, \theta = 0.99$) to a predetermined destination ($x = 6.25, y = -1.47, \theta = 0.99$). The cumulative distance from the starting home location to the designated end point is approximately 26.28 m. The objective was to optimize the route for the shortest transit time, showcasing the robot's pathfinding proficiency in a complex simulated labyrinth.

6.2.2. Simulation results

For the maze-solving task, the robots were similarly directed by the `NavFn` global planners, with the local planning algorithms tested for their path optimization efficiency. In the ROS2/Gazebo simulation, the maze-solving robot successfully navigated to the maze's center, with the `NavFn` global planner and MPPI local controller achieving the most efficient path in terms of time and distance. The Smooth MPPI and Log MPPI controllers also completed the maze with competitive times, demonstrating effective adaptability and robustness in pathfinding within the complex simulated environment. Fig. 10 and a simulation video[8] demonstrate the efficacy of the path integral controllers in navigating the maze-solving environment.

Fig. 12 illustrates the robot's orientation in a maze-solving environment. The ground truth data (dotted lines) represents the ideal orientation path for maze navigation. It is evident from the graph that the Smooth MPPI (green line) and Log MPPI (blue line) algorithms maintained a consistent orientation close to the ground truth, while the

[8] https://youtu.be/GyKDP3-NYA0.

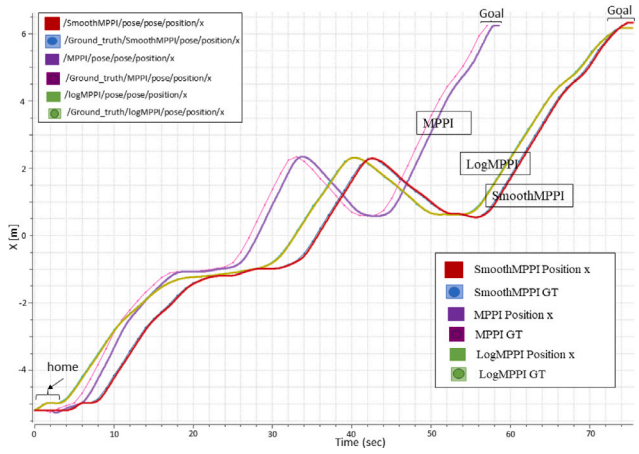

Fig. 11. Performance comparison of path integral controllers in terms of the longitudinal position (x). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

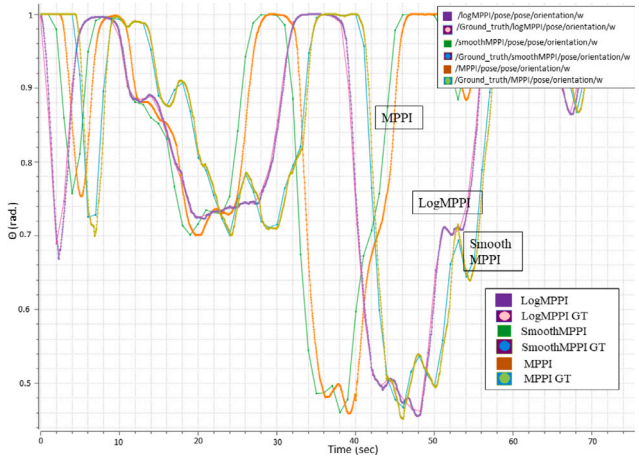

Fig. 12. Performance comparison of path integral controllers in terms of the orientation (θ). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

MPPI (red line) displayed marginally increased variance. The robot's x-position component within in maze-solving environment is shown in Fig. 11, in which the ground truth path (dotted line) is tightly followed by all three algorithms. The Smooth MPPI (red line) and Log MPPI (green line) displayed almost identical performance, with MPPI (blue line) showing slight divergence yet still within an acceptable range for effective maze navigation. In evaluating speed and time efficiency, MPPI demonstrated superior performance, achieving the goal more swiftly compared to Log MPPI and Smooth MPPI. In Fig. 11, the legend in the top left corner illustrates the ROS topics of the robot's odometry, whereas in Fig. 12, the top right corner legend presents the ROS topics associated with Gazebo ground truth data.

The simulation results demonstrate that all considered path integral algorithms MPPI, Smooth MPPI, and Log MPPI perform robustly in trajectory optimization tasks for both hotel navigation and maze-solving scenarios. The angular orientation and position tracking graphs indicate that each algorithm is capable of closely following a predetermined ground truth trajectory with minimal deviation. However, minor differences in performance suggest that certain algorithms may be more suitable for specific applications, as given in Table 4. For instance, Log MPPI and Smooth MPPI's trajectory in the hotel navigation simulation suggests a potential for finer control in more predictable environments like cafeteria or warehouse environments,

Table 4
Comparative performance metrics of different path integral controllers across different environments.

Controller	Env.	Mission time (s)	Tracking error (ℓ_∞)
MPPI	Cafe	122	0.21
	Maze	59	0.25
Smooth MPPI	Cafe	147	0.14
	Maze	75	0.25
Log MPPI	Cafe	155	0.1
	Maze	75	0.25

whereas MPPI showed promising results in the more dynamic and fast speed tracks or environments like maze-solving context or F1Tenth racing car competitions.

7. Discussion and future directions

7.1. Policy parameterization

There are multiple ways of policy parameterizations for state-feedback (Deisenroth, Neumann, Peters et al., 2013). For example, linear policies (Deisenroth, Neumann et al., 2013; Vinogradska, Bischoff, Achterhold, Koller, & Peters, 2020), radial basis function (RBF) networks (Deisenroth, Fox and Rasmussen, 2013; Deisenroth, Neumann et al., 2013; Thor, Kulvicius, & Manoonpong, 2020; Vinogradska et al., 2020), and dynamic movement primitives (DMPs) (Ijspeert, Nakanishi, Hoffmann, Pastor, & Schaal, 2013; Schaal, Peters, Nakanishi, & Ijspeert, 2005) have been commonly used for policy representations and search in robotics and control.

Note that because the PI-based control and planning algorithms presented in Section 4 are derivative-free, complex policy parameterization and optimization can be implemented without any additional effort (e.g., numerical differentiation computing gradients, Jacobians, and Hessians). This is one of the advantageous characteristics of sampling-based policy search and improvement methods such as PI.

7.2. Path integral for guided policy search

The guided policy search (GPS), first proposed in Levine and Koltun (2013), is a model-free policy-based reinforcement learning (RL). Pixel-to-torque end-to-end learning of visuomotor policies has recently become popular for RL in robotics (Levine, Finn, Darrell, & Abbeel, 2016). Compared with direct deep reinforcement learning, GPS has several advantages, such as faster convergence and better optimality. In the GPS, learning consists of two phases. The first phase involves determining a local guiding policy, in which a training set of controlled trajectories is generated. In the second phase, a complex global policy is determined via supervised learning, in which the expected KL divergence of the global policy from the guiding policy is minimized. The goal of the GPS framework is to solve an information-theoretic-constrained optimization of the following (Montgomery & Levine, 2016):

$$\begin{aligned} &\min_{\theta,\beta} \ \mathbb{E}_\beta[G(\boldsymbol{X})] \\ &\text{s.t.} \ \ D_{\mathrm{KL}}(\beta(\boldsymbol{X})\|\pi(\boldsymbol{X};\theta)) \le \epsilon \end{aligned} \tag{45}$$

where the KL divergence $D_{\mathrm{KL}}(\beta(\boldsymbol{X})\|\pi(\boldsymbol{X};\theta))$ can be rewritten as:

$$\sum_{i=0}^{N-1} \mathbb{E}_\beta[D_{\mathrm{KL}}(\beta(u_i|x_i)\|\pi(u_i|x_i;\theta))].$$

A baseline (Markovian) policy $\beta(u_i|x_i)$ is a local guiding policy used to generate sampled trajectories starting with a variety of initial conditions. A parameterized policy $\pi(u_i|x_i;\theta)$ is a high-dimensional global policy learned based on the sampled trajectories generated from the baseline policy β in a supervisory manner by minimizing the KL divergence from the local policy.

The iterative procedure for a general GPS can be summarized as follows:

(Step 1) Given $\hat{\theta}$, solve

$$\hat{\beta} = \arg\min_{\beta} \ \mathbb{E}_{\beta}[G(\boldsymbol{X})]$$
$$\text{s.t.} \ \ D_{\mathrm{KL}}(\beta(\boldsymbol{X}) \| \pi(\boldsymbol{X}; \hat{\theta})) \leq \epsilon \, .$$

(Step 2) Given $\hat{\beta}$, solve

$$\hat{\theta} = \arg\min_{\theta} \ D_{\mathrm{KL}}(\hat{\beta}(\boldsymbol{X}) \| \pi(\boldsymbol{X}; \theta)) \, .$$

(Step 3) Check convergence and repeat Steps 1 and 2 until a convergence criterion is satisfied.

Various methods have been considered to guide GPS policies. For example, gradient-based local optimal control methods such as DDP and iLQR and sampling-based local approximate optimal control methods such as PI and PI^2 can be used. Among others, we claim that because of their efficiency in exploration and fast convergence rate, PI-based sampling methods could be more appropriate as guiding policies for GPS.

7.3. Model-based reinforcement learning using PI-based policy search and optimization

In control and robotics, model-based reinforcement learning (MBRL) (Garaffa, Basso, Konzen, & de Freitas, 2021; Moerland, Broekens, Plaat, Jonker, et al., 2023; Polydoros & Nalpantidis, 2017) has been widely investigated because of the potential benefits of data efficiency, effective exploration, and enhanced stability, compared to model-free RL. It is natural to consider methods of path integral control for policy search and optimization in MBRL. The most well-known method of MBRL is the Dyna algorithm (Deisenroth & Rasmussen, 2011; Sutton, 1990) that consists of two iterative learning steps: The first step is to collect data by applying the current policy and learn dynamics model. The second step is to learn or update parameterized policies with data generated from the learned model. The second step alone is known as model-based policy optimization (MBPO) (Janner, Fu, Zhang, & Levine, 2019; Yu et al., 2020) that is particularly related to PI-based policy search and optimization. In other words, parameterized policies discussed in Sections 4.4, 4.5, and 7.1 can be improved by using sampling-based path integral control methods for MBRL and MRPO.

7.4. Sampling efficiency for variance reduction

Let us consider the importance weight defined in (38). Note that if the training policy π is optimal, then all simulated trajectories have the same weight. To measure the quality of the sampling strategy, the effective sampling size (ESS) defined as

$$\mathrm{ESS}^{\pi} = \frac{1}{\mathbb{E}_{p^{\pi}}[(\omega^{\pi})^2]} \tag{46}$$

measures the variance of the importance weights and can be used to quantify the efficiency of a sampling method (Kotecha & Djuric, 2003; Liu, 2001). A small ESS implies that the associated back-end tasks of estimation or control may result in a large variance. The most important sampling strategies suffer from decreasing ESS over time during prediction. Therefore, quantifying or approximating the ESS of a base-sampling strategy is a major problem in the application of path integral control (Thijssen & Kappen, 2015; Zhang & Chen, 2022).

7.5. Extensions to multi-agent path planning

In the literature, there are only a few studies that extend PI control to the stochastic control of multi-agent systems (MASs): path integrals for centralized control (Gómez et al., 2016), distributed control (Varnai & Dimarogonas, 2022; Wan et al., 2021a), and a two-player zero-sum stochastic differential game (SDG) (Patil, Zhou, Fridovich-Keil, & Tanaka, 2023). A linearly solvable PI control algorithm is proposed for a networked MAS in Wan, Gahlawat, Hovakimyan, Theodorou, and Voulgaris (2021b) and extended to safety-constrained cooperative control of MASs (Song et al., 2022; Song, Zhao, Wan, & Hovakimyan, 2023) using the control barrier function (CBF) in which the barrier state augmented system is defined to take care of potential conflicts between control objectives and safety requirements.

In path planning and control for multi-agent systems, it is common to assume that dynamics are independent but costs are interdependent. Consider the cost-to-go function defined for agent a as

$$G^{\bar{\pi}_a}_{a,t} = \phi_a(\bar{X}^{\bar{\pi}_a}_{a,T}) + \int_t^T L(s, \bar{X}^{\bar{\pi}_a}_{a,s}, \bar{\pi}_a(s, \bar{X}^{\bar{\pi}_a}_{a,s}))ds,$$

where $\bar{\pi}_a = (\pi_a, \pi_{\nu(a)})$ is the joint policy of the ego agent a and its neighborhood agent $\nu(a)$. Similarly the joint state and trajectory are defined as $\bar{X}_{a,s} = (X_{a,s}, X_{\nu(a),s})$ and $\bar{\boldsymbol{X}}_{a,t} = (\bar{X}_{a,s})^T_{s=t} = (X_{a,s}, X_{\nu(a),s})^T_{s=t}$.

As we have observed throughout this paper for single-agent cases, from an algorithmic point of view, the most important computation is to approximate the weights corresponding to the likelihood ratios using MC sampling methods. Similarly, policy updates or improvements in multi-agent systems can be

$$\pi_a \leftarrow \pi_a + \sum_{k=1}^{K} \iota\hat{w}^{\bar{\pi}_a}_{a,k} \delta\pi_{a,k}$$

where the probability weight is defined as

$$\iota\hat{w}^{\bar{\pi}_a}_{a,k} = \frac{\exp\left(-\frac{1}{\lambda} G^{(\pi_{a,k}, \pi_{\nu(a)})}_{a,t}\right)}{\sum_{\kappa=1}^{K} \exp\left(-\frac{1}{\lambda} G^{(\pi_{a,\kappa}, \pi_{\nu(a)})}_{a,t}\right)}$$

for which randomly perturbed policies $\pi_{a,k} = \pi_a + \delta\pi_{a,k}$ are used to simulate the controlled trajectories and compute the associated costs $G^{(\pi_{a,k}, \pi_{\nu(a)})}_{a,t}$. Here, the learning process is assumed to be asynchronous, in the sense that the policies of the neighborhood agents $\nu(a)$ are fixed when updating the policy for agent a in accordance with the simulation of the augmented trajectories $\bar{\boldsymbol{X}}_a$. Here, the individual agent's policy can be either MPC-like open-loop (feedforward) control inputs or parameterized (deterministic or stochastic) state-feedback controllers.

7.6. MPPI for trajectory optimization on manifolds

Trajectory optimization using differential geometry is very common in robotics and has been studied under manifolds such as special orthogonal and Euclidean groups SO(3) and SE(3) (Bonalli, Bylard, Cauligi, Lew & Pavone, 2019; Osa, 2022; Watterson, Liu, Sun, Smith, & Kumar, 2018, 2020). In Bonalli, Bylard et al. (2019), gradient-based sequential convex programming on manifolds was used for trajectory optimization. It was expected that many theoretical and computational frameworks for the optimization of manifolds (Boumal, 2023; Boumal, Mishra, Absil, & Sepulchre, 2014) could be applied to robotic trajectory optimization.

Applying methods of sampling-based path integral control such as MPPI to trajectory optimization on manifolds is not trivial because it requires effective accelerated approaches to generate the sampled trajectories on manifolds and sampling trajectories on manifolds with kinematic constraints are not trivial. Thus, one could employ the methods used for unscented Kalman filtering on manifolds (UKF-M) (Brossard, Barrau, & Bonnabel, 2020; Cantelobre, Chahbazian, Croux, & Bonnabel, 2020; Li, Pfaff, & Hanebeck, 2020; Menegaz, Ishihara, & Kussaba, 2018). We leave this research topic of sampling-based path integral control for trajectory optimization on manifolds for potential future work.

7.7. Motion planning for mobile robots and manipulators

While policy parameterization discussed in Section 7.1 has shown great success in robotic manipulators and locomotion control, policy

parameterization is not trivial and even not appropriate for path planning or trajectory optimization for autonomous mobile robots. This is because optimal trajectory should be determined by considering the relative pose of the robot with respect to the obstacles and the goal pose as well as the robot's ego-pose. Such relative poses can be encoded into the associated optimal control problem (OCP) as constraints and feasibility of the OCP with a parameterized policy is hard to be guaranteed for mobile robot dynamic environment. This brings us a conclusion that it is more appropriate to use the MPC-like open-loop control inputs for trajectory optimization of mobile robots, which implies that MPPI and its variations would become more successful with autonomous mobile robot navigation than PI^2-based policy search and optimization.

8. Conclusions

In this paper, we present an overview of the fundamental theoretical developments and recent advances in path integral control with a focus on sampling-based stochastic trajectory optimization. The theoretical and algorithmic frameworks of several optimal control methods employing the path integral control framework are provided, and their similarities and differences are reviewed. Python, MATLAB and ROS2/Gazebo simulation results are provided to demonstrate the effectiveness of various path integral control methods. Discussions on policy parameterization and optimization in policy search adopting path integral control, connections to model-based reinforcement learning, efficiency of sampling strategies, extending the path integral control framework to multi-agent optimal control problems, and path integral control for the trajectory optimization of manifolds are provided. We expect that sampling-based stochastic trajectory optimization employing path integral control can be applied to practical engineering problems, particularly for agile mobile robot navigation and control.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Kwang-Ki K. Kim reports financial support was provided by National Research Foundation of Korea.

Data availability

I have shared the links to my codes and data in the manuscript.

References

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

Amos, B., & Yarats, D. (2020). The differentiable cross-entropy method. In *International conference on machine learning* (pp. 291–302). PMLR.

Arouna, B. (2004). Adaptative Monte Carlo method, a variance reduction technique. *Monte Carlo Methods and Applications*, *10*(1), 1–24.

Arruda, E., Mathew, M. J., Kopicki, M., Mistry, M., Azad, M., & Wyatt, J. L. (2017). Uncertainty averse pushing with model predictive path integral control. In *2017 IEEE-RAS 17th international conference on humanoid robotics (Humanoids)* (pp. 497–502). IEEE.

Asmar, D. M., Senanayake, R., Manuel, S., & Kochenderfer, M. J. (2023). Model predictive optimized path integral strategies. In *2023 IEEE international conference on robotics and automation* ICRA, (pp. 3182–3188).

Ba, H., Fan, J., Guo, X., & Hao, J. (2021). Critic PI2: Master continuous planning via policy improvement with path integrals and deep actor-critic reinforcement learning. In *2021 6th IEEE international conference on advanced robotics and mechatronics* ICARM, (pp. 716–722). IEEE.

Balci, I. M., Bakolas, E., Vlahov, B., & Theodorou, E. A. (2022). Constrained covariance steering based Tube-MPPI. In *2022 American control conference* ACC, (pp. 4197–4202). IEEE.

Barbosa, F. S., Lacerda, B., Duckworth, P., Tumova, J., & Hawes, N. (2021). Risk-aware motion planning in partially known environments. In *2021 60th IEEE conference on decision and control* CDC, (pp. 5220–5226). IEEE.

Betts, J. T. (1998). Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, *21*(2), 193–207.

Betts, J. T. (2010). *Practical methods for optimal control and estimation using nonlinear programming*. Philadelphia, PA: SIAM.

Bonalli, R., Bylard, A., Cauligi, A., Lew, T., & Pavone, M. (2019). Trajectory optimization on manifolds: A theoretically-guaranteed embedded sequential convex programming approach. arXiv preprint arXiv:1905.07654.

Bonalli, R., Cauligi, A., Bylard, A., & Pavone, M. (2019). GuSTO: Guaranteed sequential trajectory optimization via sequential convex programming. In *2019 international conference on robotics and automation* ICRA, (pp. 6741–6747). IEEE.

Boumal, N. (2023). *An introduction to optimization on smooth manifolds*. Cambridge University Press.

Boumal, N., Mishra, B., Absil, P.-A., & Sepulchre, R. (2014). Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, *15*(1), 1455–1459.

Brossard, M., Barrau, A., & Bonnabel, S. (2020). A code for unscented Kalman filtering on manifolds (UKF-M). In *IEEE international conference on robotics and automation* ICRA, (pp. 5701–5708). IEEE.

Bugallo, M. F., Elvira, V., Martino, L., Luengo, D., Miguez, J., & Djuric, P. M. (2017). Adaptive importance sampling: The past, the present, and the future. *IEEE Signal Processing Magazine*, *34*(4), 60–79.

Cai, X., Everett, M., Sharma, L., Osteen, P. R., & How, J. P. (2022). Probabilistic traversability model for risk-aware motion planning in off-road environments. arXiv preprint arXiv:2210.00153.

Campos-Macías, L., Gómez-Gutiérrez, D., Aldana-López, R., de la Guardia, R., & Parra-Vilchis, J. I. (2017). A hybrid method for online trajectory planning of mobile robots in cluttered environments. *IEEE Robotics and Automation Letters*, *2*(2), 935–942.

Canny, J. (1988). *The complexity of robot motion planning*. Cambridge, Massachusetts: MIT Press.

Cantelobre, T., Chahbazian, C., Croux, A., & Bonnabel, S. (2020). A real-time unscented Kalman filter on manifolds for challenging AUV navigation. In *IEEE/RSJ international conference on intelligent robots and systems* IROS, (pp. 2309–2316). IEEE.

Cao, K., Cao, M., Yuan, S., & Xie, L. (2022). DIRECT: a differential dynamic programming based framework for trajectory generation. *IEEE Robotics and Automation Letters*, *7*(2), 2439–2446.

Carius, J., Ranftl, R., Farshidian, F., & Hutter, M. (2022). Constrained stochastic optimal control with learned importance sampling: A path integral approach. *International Journal of Robotics Research*, *41*(2), 189–209.

Chatzinikolaidis, I., & Li, Z. (2021). Trajectory optimization of contact-rich motions using implicit differential dynamic programming. *IEEE Robotics and Automation Letters*, *6*(2), 2626–2633.

Chen, J., Zhan, W., & Tomizuka, M. (2019). Autonomous driving motion planning with constrained iterative LQR. *IEEE Transactions on Intelligent Vehicles*, *4*(2), 244–254.

Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G. A., & Burgard, W. (2005). *Principles of robot motion: Theory, algorithms, and implementations*. Cambridge, Massachusetts: MIT Press.

Claussmann, L., Revilloud, M., Gruyer, D., & Glaser, S. (2019). A review of motion planning for highway autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, *21*(5), 1826–1848.

Costanzo, M., De Maria, G., Natale, C., & Russo, A. (2023). Modeling and control of sampled-data image-based visual servoing with three-dimensional features. *IEEE Transactions on Control Systems Technology*, (Early Access).

De Boer, P.-T., Kroese, D. P., Mannor, S., & Rubinstein, R. Y. (2005). A tutorial on the cross-entropy method. *Annals of Operations Research*, *134*(1), 19–67.

Dearing, T. L., Hauser, J., Chen, X., Nicotra, M. M., & Petersen, C. (2022). Efficient trajectory optimization for constrained spacecraft attitude maneuvers. *Journal of Guidance, Control, and Dynamics*, *45*(4), 638–650.

Deisenroth, M. P., Fox, D., & Rasmussen, C. E. (2013). Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(2), 408–423.

Deisenroth, M. P., Neumann, G., Peters, J., et al. (2013). A survey on policy search for robotics. *Foundations and Trends® in Robotics*, *2*(1–2), 1–142.

Deisenroth, M., & Rasmussen, C. E. (2011). PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th international conference on machine learning* ICML-11, (pp. 465–472).

Domahidi, A., & Jerez, J. (2014–2023). *FORCES professional*. EmbotechAG, url=https://embotech.com/FORCES-Pro.

Elbanhawi, M., & Simic, M. (2014). Sampling-based robot motion planning: A review. *IEEE Access*, *2*, 56–77.

Eysenbach, B., & Levine, S. (2019). If MaxEnt RL is the answer, what is the question? arXiv preprint arXiv:1910.01913.

Faessler, M., Franchi, A., & Scaramuzza, D. (2017). Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE Robotics and Automation Letters*, *3*(2), 620–626.

Fleming, W. H., & McEneaney, W. M. (1995). Risk-sensitive control on an infinite time horizon. *SIAM Journal on Control and Optimization*, *33*(6), 1881–1915.

Fleming, W. H., & Rishel, R. W. (2012). *Deterministic and dtochastic optimal control*: *vol. 1*, New York: Springer Science & Business Media.

Fleming, W. H., & Soner, H. M. (2006). *Controlled Markov processes and viscosity solutions*: *vol. 25*, New York: Springer Science & Business Media.

Fu, J., Li, C., Teng, X., Luo, F., & Li, B. (2020). Compound heuristic information guided policy improvement for robot motor skill acquisition. *Applied Sciences*, *10*(15), 5346.

Gammell, J. D., & Strub, M. P. (2020). Asymptotically optimal sampling-based motion planning methods. arXiv preprint arXiv:2009.10484.

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

Garaffa, L. C., Basso, M., Konzen, A. A., & de Freitas, E. P. (2021). Reinforcement learning for mobile robotics exploration: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, *34*(8), 3796–3810.

Garcia, I., & How, J. P. (2005). Trajectory optimization for satellite reconfiguration maneuvers with position and attitude constraints. In *Proceedings of the 2005, American control conference, 2005* (pp. 889–894). IEEE.

Gatherer, A., & Manchester, Z. (2019). Magnetorquer-only attitude control of small satellites using trajectory optimization. In *Proceedings of AAS/AIAA astrodynamics specialist conference*.

Gómez, V., Kappen, H. J., Peters, J., & Neumann, G. (2014). Policy search for path integral control. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 482–497). Springer.

Gómez, V., Thijssen, S., Symington, A., Hailes, S., & Kappen, H. J. (2016). Real-time stochastic optimal control for multi-agent quadrotor systems. In *Proceedings of the twenty-sixth international conference on automated planning and scheduling* ICAPS, (pp. 468–476).

Ha, J.-S., Park, S.-S., & Choi, H.-L. (2019). Topology-guided path integral approach for stochastic optimal control in cluttered environment. *Robotics and Autonomous Systems*, *113*, 81–93.

Haarnoja, T., Tang, H., Abbeel, P., & Levine, S. (2017). Reinforcement learning with deep energy-based policies. In *International conference on machine learning* (pp. 1352–1361). PMLR.

Han, Z., Wang, Z., Pan, N., Lin, Y., Xu, C., & Gao, F. (2021). Fast-Racing: An open-source strong baseline for SE(3) planning in autonomous drone racing. *IEEE Robotics and Automation Letters*, *6*(4), 8631–8638.

Hanover, D., Loquercio, A., Bauersfeld, L., Romero, A., Penicka, R., Song, Y., et al. (2023). Autonomous drone racing: A survey. arXiv e-prints, pp. arXiv–2301.

Hansen, N. (2016). The CMA evolution strategy: A tutorial. arXiv preprint arXiv:1604.00772.

Hansen, N., & Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, *9*(2), 159–195.

Higgins, J., Mohammad, N., & Bezzo, N. (2023). A model predictive path integral method for fast, proactive, and uncertainty-aware UAV planning in cluttered environments. arXiv preprint arXiv:2308.00914.

Hou, L., Wang, H., Zou, H., & Zhou, Y. (2022). Robotic manipulation planning for automatic peeling of glass substrate based on online learning model predictive path integral. *Sensors*, *22*(3), 1292.

Houghton, M. D., Oshin, A. B., Acheson, M. J., Theodorou, E. A., & Gregory, I. M. (2022). Path planning: Differential dynamic programming and model predictive path integral control on VTOL aircraft. In *AIAA SCITECH 2022 forum* (p. 0624).

Howell, T. A., Jackson, B. E., & Manchester, Z. (2019). ALTRO: A fast solver for constrained trajectory optimization. In *2019 IEEE/RSJ international conference on intelligent robots and systems* IROS, (pp. 7674–7679). IEEE.

Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., & Schaal, S. (2013). Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation*, *25*(2), 328–373.

Ijspeert, A., Nakanishi, J., & Schaal, S. (2002). Learning attractor landscapes for learning motor primitives. *Advances in Neural Information Processing Systems*, *15*.

Jacobson, D. H., & Mayne, D. Q. (1970). *Differential dynamic programming*. new York: Elsevier Publishing Company, no. 24.

Janner, M., Fu, J., Zhang, M., & Levine, S. (2019). When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, *32*, 1–9.

Janson, L., Ichter, B., & Pavone, M. (2018). Deterministic sampling-based motion planning: Optimality, complexity, and performance. *International Journal of Robotics Research*, *37*(1), 46–61.

Kalakrishnan, M., Chitta, S., Theodorou, E., Pastor, P., & Schaal, S. (2011). STOMP: Stochastic trajectory optimization for motion planning. In *IEEE international conference on robotics and automation* ICRA, (pp. 4569–4574). IEEE.

Kappen, H. J. (2005). Linear theory for control of nonlinear stochastic systems. *Physical Review Letters*, *95*(20), Article 200201.

Kappen, H. J. (2007). An introduction to stochastic control theory, path integrals and reinforcement learning. In *AIP conference proceedings: vol. 887*, (pp. 149–181). American Institute of Physics, no. 1.

Kappen, H. (2011). Optimal control theory and the linear bellman equation. In D. Barber, A. T. Cemgil, & S. Chiappa (Eds.), *Bayesian time series models* (pp. 363–387). Cambridge: Cambridge University Press.

Kappen, H. J., Gómez, V., & Opper, M. (2012). Optimal control as a graphical model inference problem. *Machine Learning*, *87*(2), 159–182.

Kappen, H. J., & Ruiz, H. C. (2016). Adaptive importance sampling for control and inference. *Journal of Statistical Physics*, *162*(5), 1244–1266.

Kelly, M. (2017). An introduction to trajectory optimization: How to do your own direct collocation. *SIAM Review*, *59*(4), 849–904.

Kiani, F., Seyyedabbasi, A., Aliyev, R., Gulle, M. U., Basyildiz, H., & Shah, M. A. (2021). Adapted-RRT: Novel hybrid method to solve three-dimensional path planning problem using sampling and metaheuristic-based algorithms. *Neural Computing and Applications*, *33*(22), 15569–15599.

Kim, M.-G., & Kim, K.-K. K. (2022a). An extension of interior point differential dynamic programming for optimal control problems with second-order conic constraints. *Transactions of the Korean Institute of Electrical Engineers*, *71*(11), 1666–1672.

Kim, M.-G., & Kim, K.-K. K. (2022b). MPPI-IPDDP: Hybrid method of collision-free smooth trajectory generation for autonomous robots. arXiv preprint arXiv:2208.02439.

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

Kingston, Z., Moll, M., & Kavraki, L. E. (2018). Sampling-based methods for motion planning with constraints. *Annual Review of Control, Robotics, and Autonomous Systems*, *1*(1), 159–185.

Kobilarov, M. (2012). Cross-entropy motion planning. *International Journal of Robotics Research*, *31*(7), 855–871.

Kotecha, J. H., & Djuric, P. M. (2003). Gaussian sum particle filtering. *IEEE Transactions on Signal Processing*, *51*(10), 2602–2612.

Kuffner, J. J., & LaValle, S. M. (2000). RRT-connect: An efficient approach to single-query path planning. *vol. 2*, In *2000 IEEE international conference on robotics and automation* ICRA, (pp. 995–1001). IEEE.

Kupcsik, A., Deisenroth, M., Peters, J., & Neumann, G. (2013). Data-efficient contextual policy search for robot movement skills. In *Proceedings of the national conference on artificial intelligence*. AAAI, Bellevue.

Kwon, H.-H., & Choi, H.-L. (2020). A convex programming approach to mid-course trajectory optimization for air-to-ground missiles. *International Journal of Aeronautical and Space Sciences*, *21*, 479–492.

Lambert, A., Fishman, A., Fox, D., Boots, B., & Ramos, F. (2020). Stein variational model predictive control. arXiv preprint arXiv:2011.07641.

Lan, M., Lai, S., Lee, T. H., & Chen, B. M. (2021). A survey of motion and task planning techniques for unmanned multicopter systems. *Unmanned Systems*, *9*(02), 165–198.

Latombe, J.-C. (2012). *Robot motion planning: vol. 124*, New York: Springer Science & Business Media.

LaValle, S. M. (2006). *Planning algorithms*. New York: Cambridge University Press.

Lefebvre, T., & Crevecoeur, G. (2019). Path integral policy improvement with differential dynamic programming. In *2019 IEEE/ASME international conference on advanced intelligent mechatronics* AIM, (pp. 739–745). IEEE.

Lefebvre, T., & Crevecoeur, G. (2022). Entropy regularised deterministic optimal control: From path integral solution to sample-based trajectory optimisation. In *2022 IEEE/ASME international conference on advanced intelligent mechatronics* AIM, (pp. 401–408). IEEE.

Levine, S. (2018). Reinforcement learning and control as probabilistic inference: Tutorial and review. arXiv preprint arXiv:1805.00909.

Levine, S., Finn, C., Darrell, T., & Abbeel, P. (2016). End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, *17*(1), 1334–1373.

Levine, S., & Koltun, V. (2013). Guided policy search. In *International conference on machine learning* (pp. 1–9). PMLR.

Li, K., Pfaff, F., & Hanebeck, U. D. (2020). Unscented dual quaternion particle filter for SE(3) estimation. *IEEE Control Systems Letters*, *5*(2), 647–652.

Likhachev, M. (2010). Search-based planning lab. [Online]. Available: http://sbpl.net/Home.

Liptser, R. S., & Shiriaev, A. N. (2001a). *Statistics of random processes I: General theory* (2nd ed.). Heidelberg: Springer-Verlag Berlin.

Liptser, R. S., & Shiriaev, A. N. (2001b). *Statistics of random processes II: Applications* (2nd ed.). Heidelberg: Springer-Verlag Berlin.

Liu, J. S. (2001). *Monte Carlo strategies in scientific computing: vol. 75*, New York: Springer.

Macenski, S., Foote, T., Gerkey, B., Lalancette, C., & Woodall, W. (2022). Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, *7*(66), eabm6074.

Macenski, S., Moore, T., Lu, D. V., Merzlyakov, A., & Ferguson, M. (2023). From the desks of ROS maintainers: A survey of modern & capable mobile robotics algorithms in the robot operating system 2. *Robotics and Autonomous Systems*, Article 104493.

Malyuta, D., Reynolds, T. P., Szmuk, M., Lew, T., Bonalli, R., Pavone, M., et al. (2022). < Convex optimization for trajectory generation: A tutorial on generating dynamically feasible trajectories reliably and efficiently. *IEEE Control Systems Magazine*, *42*(5), 40–113.

Malyuta, D., Yu, Y., Elango, P., & Açıkmeşe, B. (2021). Advances in trajectory optimization for space vehicle control. *Annual Reviews in Control*, *52*, 282–315.

Manyam, S. G., Casbeer, D. W., Weintraub, I. E., & Taylor, C. (2021). Trajectory optimization for rendezvous planning using quadratic Bézier curves. In *2021 IEEE/RSJ international conference on intelligent robots and systems* IROS, (pp. 1405–1412). IEEE.

Martino, L., Elvira, V., Luengo, D., & Corander, J. (2015). An adaptive population importance sampler: Learning from uncertainty. *IEEE Transactions on Signal Processing*, *63*(16), 4422–4437.

Mayne, D. Q. (1973). Differential dynamic programming–a unified approach to the optimization of dynamic systems. In *Control and dynamic systems: vol. 10*, (pp. 179–254). Elsevier.

Menegaz, H. M., Ishihara, J. Y., & Kussaba, H. T. (2018). Unscented Kalman filters for Riemannian state-space systems. *IEEE Transactions on Automatic Control*, *64*(4), 1487–1502.

Moerland, T. M., Broekens, J., Plaat, A., Jonker, C. M., et al. (2023). Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, *16*(1), 1–118.

Mohamed, I. S. (2021). MPPI-VS: Sampling-based model predictive control strategy for constrained image-based and position-based visual servoing. arXiv preprint arXiv:2104.04925.

Mohamed, I. S., Ali, M., & Liu, L. (2023). GP-guided MPPI for efficient navigation in complex unknown cluttered environments. arXiv preprint arXiv:2307.04019.

Mohamed, I. S., Allibert, G., & Martinet, P. (2020). 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, (pp. 196–203). IEEE.

Mohamed, I. S., Allibert, G., & Martinet, P. (2021). Sampling-based MPC for constrained vision based control. In *2021 IEEE/RSJ international conference on intelligent robots and systems* IROS, (pp. 3753–3758). IEEE.

Mohamed, I. S., Yin, K., & Liu, L. (2022). Autonomous navigation of AGVs in unknown cluttered environments: log-MPPI control strategy. *IEEE Robotics and Automation Letters*, *7*(4), 10240–10247.

Montgomery, W. H., & Levine, S. (2016). Guided policy search via approximate mirror descent. *Advances in Neural Information Processing Systems (NIPS)*, *29*.

Neve, T., Lefebvre, T., & Crevecoeur, G. (2022). Comparative study of sample based model predictive control with application to autonomous racing. In *2022 IEEE/ASME international conference on advanced intelligent mechatronics* AIM, (pp. 1632–1638). IEEE.

Nicolay, P., Petillot, Y., Marfeychuk, M., Wang, S., & Carlucho, I. (2023). Enhancing AUV autonomy with model predictive path integral control. arXiv preprint arXiv:2308.05547.

Okada, M., Aoshima, T., & Rigazio, L. (2017). Path integral networks: End-to-end differentiable optimal control. arXiv preprint arXiv:1706.09597.

Oksendal, B. (2013). *Stochastic differential equations: An introduction with applications*. New York: Springer Science & Business Media.

Osa, T. (2022). Motion planning by learning the solution manifold in trajectory optimization. *International Journal of Robotics Research*, *41*(3), 281–311.

Paden, B., Čáp, M., Yong, S. Z., Yershov, D., & Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, *1*(1), 33–55.

Pan, Y., Theodorou, E., & Kontitsis, M. (2015). Sample efficient path integral control under uncertainty. *Advances in Neural Information Processing Systems*, *28*.

Park, J., Kim, I., Suk, J., & Kim, S. (2023). Trajectory optimization for takeoff and landing phase of UAM considering energy and safety. *Aerospace Science and Technology*, *140*, Article 108489.

Patil, A., Zhou, Y., Fridovich-Keil, D., & Tanaka, T. (2023). Risk-minimizing two-player zero-sum stochastic differential game via path integral control. arXiv preprint arXiv:2308.11546.

Pavlov, A., Shames, I., & Manzie, C. (2021). Interior point differential dynamic programming. *IEEE Transactions on Control Systems Technology*, *29*(6), 2720–2727.

Polydoros, A. S., & Nalpantidis, L. (2017). Survey of model-based reinforcement learning: Applications on robotics. *Journal of Intelligent and Robotic Systems*, *86*(2), 153–173.

Pourchot, A., & Sigaud, O. (2018). CEM-RL: Combining evolutionary and gradient-based methods for policy search. arXiv preprint arXiv:1810.01222.

Pradeep, P., Lauderdale, T. A., Chatterji, G. B., Sheth, K., Lai, C. F., Sridhar, B., et al. (2020). Wind-optimal trajectories for multirotor eVTOL aircraft on UAM missions. In *Aiaa aviation 2020 forum* (p. 3271).

Pravitra, J., Ackerman, K. A., Cao, C., Hovakimyan, N., & Theodorou, E. A. (2020). L1-Adaptive MPPI architecture for robust and agile control of multirotors. In *2020 IEEE/RSJ international conference on intelligent robots and systems* IROS, (pp. 7661–7666).

Pravitra, J., Theodorou, E., & Johnson, E. N. (2021). Flying complex maneuvers with model predictive path integral control. In *AIAA scitech 2021 forum* (p. 1957).

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

Rao, A. V. (2014). Trajectory optimization: A survey. In *Optimization and optimal control in automotive systems* (pp. 3–21). Springer.

Ravankar, A. A., Ravankar, A., Emaru, T., & Kobayashi, Y. (2020). HPPRM: Hybrid potential based probabilistic roadmap algorithm for improved dynamic path planning of mobile robots. *IEEE Access*, *8*, 221743–221766.

Roh, H., Oh, Y.-J., Tahk, M.-J., Kwon, K.-J., & Kwon, H.-H. (2020). L1 penalized sequential convex programming for fast trajectory optimization: With application to optimal missile guidance. *International Journal of Aeronautical and Space Sciences*, *21*, 493–503.

Rubinstein, R. Y., & Kroese, D. P. (2004). *The cross-entropy method: A unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*: *vol. 133*, New York: Springer-Verlag.

Rubinstein, R. Y., & Kroese, D. P. (2016). *Simulation and the Monte Carlo method* (2nd ed.). New York: John Wiley & Sons.

Ruiz, H.-C., & Kappen, H. J. (2017). Particle smoothing for hidden diffusion processes: Adaptive path integral smoother. *IEEE Transactions on Signal Processing*, *65*(12), 3191–3203.

Särkkä, S. (2008). Unscented rauch–tung–striebel smoother. *IEEE Transactions on Automatic Control*, *53*(3), 845–849.

Schaal, S., Peters, J., Nakanishi, J., & Ijspeert, A. (2005). Learning movement primitives. In *Robotics research. the eleventh international symposium* (pp. 561–572). Springer.

Song, Y., Steinweg, M., Kaufmann, E., & Scaramuzza, D. (2021). Autonomous drone racing with deep reinforcement learning. In *2021 IEEE/RSJ international conference on intelligent robots and systems* IROS, (pp. 1205–1212). IEEE.

Song, L., Wan, N., Gahlawat, A., Tao, C., Hovakimyan, N., & Theodorou, E. A. (2022). Generalization of safe optimal control actions on networked multiagent systems. *IEEE Transactions on Control of Network Systems*, *10*(1), 491–502.

Song, L., Zhao, P., Wan, N., & Hovakimyan, N. (2023). Safety embedded stochastic optimal control of networked multi-agent systems via barrier states. In *2023 American control conference* ACC, (pp. 2554–2559). IEEE.

Stich, S. U., Raj, A., & Jaggi, M. (2017). Safe adaptive importance sampling. *Advances in Neural Information Processing Systems*, *30*.

Stulp, F., & Sigaud, O. (2012a). Path integral policy improvement with covariance matrix adaptation. arXiv preprint arXiv:1206.4621.

Stulp, F., & Sigaud, O. (2012b). Policy improvement methods: Between black-box optimization and episodic reinforcement learning. *HAL Open Science*.

Sucan, I. A., Moll, M., & Kavraki, L. E. (2012). The open motion planning library. *IEEE Robotics & Automation Magazine*, *19*(4), 72–82.

Sun, S., Romero, A., Foehn, P., Kaufmann, E., & Scaramuzza, D. (2022). A comparative study of nonlinear MPC and differential-flatness-based control for quadrotor agile flight. *IEEE Transactions on Robotics*, *38*(6), 3357–3373.

Sutton, R. S. (1990). Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990* (pp. 216–224). Elsevier.

Tao, C., Kim, H., & Hovakimyan, N. (2023). RRT guided model predictive path integral method. arXiv preprint arXiv:2301.13143.

Tao, C., Yoon, H.-J., Kim, H., Hovakimyan, N., & Voulgaris, P. (2022). Path integral methods with stochastic control barrier functions. In *2022 IEEE 61st conference on decision and control* CDC, (pp. 1654–1659). IEEE.

Teng, S., Hu, X., Deng, P., Li, B., Li, Y., Ai, Y., et al. (2023). Motion planning for autonomous driving: The state of the art and future perspectives. *IEEE Transactions on Intelligent Vehicles*.

Testouri, M., Elghazaly, G., & Frank, R. (2023). Towards a safe real-time motion planning framework for autonomous driving systems: An MPPI approach. arXiv preprint arXiv:2308.01654.

Thalmeier, D., Kappen, H. J., Totaro, S., & Gómez, V. (2020). Adaptive smoothing for path integral control. *Journal of Machine Learning Research*, *21*(1), 7814–7850.

Theodorou, E. A. (2011). *Iterative path integral stochastic optimal control: Theory and applications to motor control* (Ph.D. dissertation), University of Southern California.

Theodorou, E. A. (2015). Nonlinear stochastic control and information theoretic dualities: Connections, interdependencies and thermodynamic interpretations. *Entropy*, *17*(5), 3352–3375.

Theodorou, E., Buchli, J., & Schaal, S. (2010). A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, *11*, 3137–3181.

Theodorou, E. A., & Todorov, E. (2012). Relative entropy and free energy dualities: Connections to path integral and KL control. In *2012 51st IEEE conference on decision and control* CDC, (pp. 1466–1473). IEEE.

Thijssen, S. A. (2016). *Path integral control* (Ph.D. dissertation), Radboud University.

Thijssen, S., & Kappen, H. (2015). Path integral control and state-dependent feedback. *Physical Review E*, *91*(3), Article 032104.

Thijssen, S., & Kappen, H. (2018). Consistent adaptive multiple importance sampling and controlled diffusions. arXiv preprint arXiv:1803.07966.

Thor, M., Kulvicius, T., & Manoonpong, P. (2020). Generic neural locomotion control framework for legged robots. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(9), 4013–4025.

Van Den Broek, B., Wiegerinck, W., & Kappen, B. (2008). Graphical model inference in optimal control of stochastic multi-agent systems. *Journal of Artificial Intelligence Research*, *32*, 95–122.

Varnai, P., & Dimarogonas, D. V. (2020). Path integral policy improvement: An information-geometric optimization approach. [Online]. Available: 10.13140/RG.2.2.13969.76645.

Varnai, P., & Dimarogonas, D. V. (2022). Multi-agent stochastic control using path integral policy improvement. In *American control conference* ACC, (pp. 3406–3411). IEEE.

Vinogradska, J., Bischoff, B., Achterhold, J., Koller, T., & Peters, J. (2020). Numerical quadrature for probabilistic policy search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(1), 164–175.

Von Stryk, O., & Bulirsch, R. (1992). Direct and indirect methods for trajectory optimization. *Annals of Operations Research*, *37*(1), 357–373.

Wan, N., Gahlawat, A., Hovakimyan, N., Theodorou, E. A., & Voulgaris, P. G. (2021a). Cooperative path integral control for stochastic multi-agent systems. In *American control conference* ACC, (pp. 1262–1267). IEEE.

Wan, N., Gahlawat, A., Hovakimyan, N., Theodorou, E. A., & Voulgaris, P. G. (2021b). Distributed algorithms for linearly-solvable optimal control in networked multi-agent systems. arXiv preprint arXiv:2102.09104.

Wang, M., Diepolder, J., Zhang, S., Söpper, M., & Holzapfel, F. (2021). Trajectory optimization-based maneuverability assessment of eVTOL aircraft. *Aerospace Science and Technology*, *117*, Article 106903.

Wang, Z., So, O., Lee, K., & Theodorou, E. A. (2021). Adaptive risk sensitive model predictive control with stochastic search. In *Learning for dynamics and control* (pp. 510–522). PMLR.

Watson, J., Abdulsamad, H., & Peters, J. (2020). Stochastic optimal control as approximate input inference. In *Conference on robot learning* (pp. 697–716). PMLR.

Watterson, M., Liu, S., Sun, K., Smith, T., & Kumar, V. (2018). Trajectory optimization on manifolds with applications to SO(3) and $\mathbb{R}^3 \times \mathbb{S}^2$. In *Robotics: Science and systems* (p. 9).

Watterson, M., Liu, S., Sun, K., Smith, T., & Kumar, V. (2020). Trajectory optimization on manifolds with applications to quadrotor systems. *International Journal of Robotics Research*, *39*(2–3), 303–320.

Weiss, A., Leve, F., Baldwin, M., Forbes, J. R., & Kolmanovsky, I. (2014). Spacecraft constrained attitude control using positively invariant constraint admissible sets on $SO(3) \times \mathbb{R}^3$. In *2014 American control conference* (pp. 4955–4960). IEEE.

Wen, M., & Topcu, U. (2018). Constrained cross-entropy method for safe reinforcement learning. *Advances in Neural Information Processing Systems*, *31*.

Whittle, P. (1991). Likelihood and cost as path integrals. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *53*(3), 505–529.

Williams, G. R. (2019). *Model predictive path integral control: Theoretical foundations and applications to autonomous driving* (Ph.D. dissertation), Georgia Institute of Techonology.

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

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

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

Williams, G., Rombokas, E., & Daniel, T. (2015). GPU based path integral control with learned dynamics. arXiv preprint arXiv:1503.00330.

Xie, Z., Liu, C. K., & Hauser, K. (2017). Differential dynamic programming with non-linear constraints. In *2017 IEEE international conference on robotics and automation* ICRA, (pp. 695–702). IEEE.

Yamamoto, K., Ariizumi, R., Hayakawa, T., & Matsuno, F. (2020). Path integral policy improvement with population adaptation. *IEEE Transactions on Cybernetics*, *52*(1), 312–322.

Yin, J., Dawson, C., Fan, C., & Tsiotras, P. (2023). Shield model predictive path integral: A computationally efficient robust MPC approach using control barrier functions. arXiv preprint arXiv:2302.11719.

Yin, J., Zhang, Z., Theodorou, E., & Tsiotras, P. (2022). Trajectory distribution control for model predictive path integral control using covariance steering. In *2022 international conference on robotics and automation* ICRA, (pp. 1478–1484). IEEE.

Yin, J., Zhang, Z., & Tsiotras, P. (2023). Risk-aware model predictive path integral control using conditional value-at-risk. In *2023 IEEE international conference on robotics and automation* ICRA, (pp. 7937–7943). IEEE.

Yu, Z., Si, Z., Li, X., Wang, D., & Song, H. (2022). A novel hybrid particle swarm optimization algorithm for path planning of UAVs. *IEEE Internet of Things Journal*, *9*(22), 22547–22558.

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J. Y., Levine, S., et al. (2020). MOPO: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, *33*, 14129–14142.

Zeng, J., Zhang, B., & Sreenath, K. (2021). Safety-critical model predictive control with discrete-time control barrier function. In *2021 American control conference* ACC, (pp. 3882–3889). IEEE.

Zhang, Q., & Chen, Y. (2022). Path integral sampler: A stochastic control approach for sampling. In *International conference on learning representations*.

Zhang, Z., Jin, J., Jagersand, M., Luo, J., & Schuurmans, D. (2022). A simple decentralized cross-entropy method. *Advances in Neural Information Processing Systems*, *35*, 36495–36506.

Zhang, W., Wang, H., Hartmann, C., Weber, M., & Schute, C. (2014). Applications of the cross-entropy method to importance sampling and optimal control of diffusions. *SIAM Journal on Scientific Computing*, *36*(6), A2654–A2672.

Zhong, X., Tian, J., Hu, H., & Peng, X. (2020). Hybrid path planning based on safe A* algorithm and adaptive window approach for mobile robot in large-scale dynamic environment. *Journal of Intelligent and Robotic Systems*, *99*(1), 65–77.

Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., et al. (2013). Chomp: Covariant hamiltonian optimization for motion planning. *International Journal of Robotics Research*, *32*(9–10), 1164–1193.