Information-Theoretic Stochastic Optimal Control via Incremental Sampling-based Algorithms

Oktay Arslan
oktay@gatech.edu

Evangelos A. Theodorou
evangelos.theodorou@ae.gatech.edu

Panagiotis Tsiotras
tsiotras@gatech.edu

***Abstract*—This paper considers optimal control of dynamical systems which are represented by nonlinear stochastic differential equations. It is well-known that the optimal control policy for this problem can be obtained as a function of a value function that satisfies a nonlinear partial differential equation, namely, the Hamilton-Jacobi-Bellman equation. This nonlinear PDE must be solved backwards in time, and this computation is intractable for large scale systems. Under certain assumptions, and after applying a logarithmic transformation, an alternative characterization of the optimal policy can be given in terms of a path integral. Path Integral (PI) based control methods have recently been shown to provide elegant solutions to a broad class of stochastic optimal control problems. One of the implementation challenges with this formalism is the computation of the expectation of a cost functional over the trajectories of the unforced dynamics. Computing such expectation over trajectories that are sampled uniformly may induce numerical instabilities due to the exponentiation of the cost. Therefore, sampling of low-cost trajectories is essential for the practical implementation of PI-based methods. In this paper, we use incremental sampling-based algorithms to sample useful trajectories from the unforced system dynamics, and make a novel connection between Rapidly-exploring Random Trees (RRTs) and information-theoretic stochastic optimal control. We show the results from the numerical implementation of the proposed approach to several examples.**

Keywords*—*path integral, stochastic optimal control, sampling-based algorithms

I. Introduction

In [1], [2], the authors showed the connection between Kullback-Leibler (KL) and Path Integral (PI) control with an information-theoretic view of stochastic optimal control. In addition, the authors derived the iterative path integral optimal control without relying on policy parameterizations, as in [3]. We review the work in [1], [2] starting with the definitions of free energy and relative entropy and their connections to dynamic programming. In addition, we discuss how the iterative scheme developed in [2] and [1] can be modified to incorporate incremental sampling-based methods such as Rapidly-exploring Random Trees (RRT) to guide sampling [4]–[6]. An early approach which leverages the RRT algorithm to solve stochastic optimal control problems for linear systems under environmental uncertainty is given in [7].

Within the mathematical framework of path integral control, the Feynman-Kac lemma plays an essential role, since it creates a connection between Stochastic Differential Equations (SDEs) and backward Partial Differential Equations (PDEs). This fundamental connection between SDEs and backward PDEs has inspired new avenues for the development of stochastic control algorithms such as Policy Improvement with Path Integrals (PI^2) [3] that rely on forward sampling. PI^2 has been applied to a plethora of motor control tasks from robotic object manipulation and locomotion to general trajectory optimization and gain scheduling [3], [8], [9], but it relies on a suitable parameterization of the optimal control policy. While policy parameterization such as Dynamic Movement Primitives (DMPs) [10] improves sampling by steering trajectories in high-dimensional state spaces towards areas of interest, it does not exploit the feedback structure provided by the path integral control framework. In PI^2 trajectories are sampled from the initial state of the task, the optimal parameter variations are computed, and the parameters are updated. In the next iteration, trajectories are sampled again from the same initial state and the iterative process continues until convergence. It is clear that in the case of policy parameterization one has to explicitly design the structure of the feedback control policy and then treat the gains as parameters to be optimized.

With respect to information theoretic formulations of policy search methods, [11], our work here does not depend on policy parameterizations and is grounded on the Relative Entropy - Free Energy Dualities and their connection to Dynamic Programming Principle.

The authors are with the Daniel Guggenheim School of Aerospace Engineering and the Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA.

II. Notation

A *probability space* is a triple $(\Omega, \mathcal{F}, \mathsf{p})$ where $(\Omega, \mathcal{F})$ is a measurable space with Ω a non-empty set, which is called the *sample space*, $\mathcal{F} \subseteq 2^{\Omega}$ a σ-algebra of subsets of Ω, whose elements are called events, and p is a *probability measure* on $\mathcal{F}$, that is, p is a finite measure on $\mathcal{F}$ with $\mathsf{p}(\Omega) = 1$.

A real random variable is a function $X : \Omega \to \mathbb{R}$ with the property that $\{\omega \in \Omega : X(\omega) \leq x\} \in \mathcal{F}$ for each $x \in \mathbb{R}$. Such a function is said to be $\mathcal{F}$-measurable. An extended (real) random variable can also take the values $\pm\infty$. If X is a random variable on the probability space $(\Omega, \mathcal{F}, \mathsf{p})$, then its *expectation* is defined by

$$\mathbb{E}_{\mathsf{p}}\left[X\right] = \int_{\Omega} X(\omega)\, \mathrm{d}\mathsf{p}(\omega), \tag{1}$$

provided that the integral in the right-hand side exists. As usual, and for notational simplicity, in the sequel we will drop the explicit dependence on $\omega \in \Omega$ in (1). In other words, the notation $\mathbb{E}_{\mathsf{p}}\left[X\right]$ is another (shorter) notation for the integral $\int X \mathrm{d}\mathsf{p}$.

III. Stochastic Control Based on Free Energy and Relative Entropy Dualities

Let $(\Omega, \mathcal{F})$ be a measurable space where Ω is a non-empty set and $\mathcal{F} \subseteq 2^{\Omega}$ is a σ-algebra of subsets of Ω, and let $\mathbf{P}(\Omega)$ be the set of all probability measures defined on $(\Omega, \mathcal{F})$.

Definition 1: Let $\mathsf{p} \in \mathbf{P}(\Omega)$ be a probability measure, $\mathbf{x} = \mathbf{x}(\omega),\ \omega \in \Omega$ be a random variable, $t, \rho \in \mathbb{R}$ be real numbers, and let $\mathcal{J}(\mathbf{x}, t)$ be a measurable function. The *Helmholtz free energy* of $\mathcal{J}(\mathbf{x}, t)$ with respect to p is defined by

$$\begin{aligned}\mathcal{E}_{\mathsf{p}}\left(\mathcal{J}(\mathbf{x}, t); \rho\right) &= \log\left(\int \exp\left(\rho \mathcal{J}(\mathbf{x}, t)\right) \mathrm{d}\mathsf{p}\right) \\ &= \log \mathbb{E}_{\mathsf{p}}\left[\exp\left(\rho \mathcal{J}(\mathbf{x}, t)\right)\right]. \qquad (2)\end{aligned}$$

Definition 2: Let $\mathsf{p}, \mathsf{q} \in \mathbf{P}(\Omega)$ be two probability measures. The *relative entropy* of p with respect to q is defined as[1]:

$$\mathbb{KL}\left(\mathsf{q} \| \mathsf{p}\right) = \begin{cases} \int \log\left(\frac{\mathrm{d}\mathsf{q}}{\mathrm{d}\mathsf{p}}\right) \mathrm{d}\mathsf{q} & \text{if } \mathsf{q} \ll \mathsf{p} \text{ and } \log\left(\frac{\mathrm{d}\mathsf{q}}{\mathrm{d}\mathsf{p}}\right) \in L^1, \\ +\infty & \text{otherwise.} \end{cases} \qquad (3)$$

We will also consider the function $\xi(\mathbf{x}, t)$, defined by

$$\xi(\mathbf{x}, t) = \tfrac{1}{\rho} \mathcal{E}_{\mathsf{p}}\left(\mathcal{J}(\mathbf{x}, t); \rho\right) = \tfrac{1}{\rho} \log \mathbb{E}_{\mathsf{p}}\left[\exp\left(\rho \mathcal{J}(\mathbf{x}, t)\right)\right]. \quad (4)$$

To derive the basic relationship between free energy and relative entropy [13], we express the expectation $\mathbb{E}_{\mathsf{p}}$ taken under the probability measure p as a function of the expectation $\mathbb{E}_{\mathsf{q}}$ taken under the probability measure q. More precisely, we have:

$$\mathbb{E}_{\mathsf{p}}\left[\exp\left(\rho \mathcal{J}(\mathbf{x}, t)\right)\right] = \int \exp\left(\rho \mathcal{J}(\mathbf{x}, t)\right) \frac{\mathrm{d}\mathsf{p}}{\mathrm{d}\mathsf{q}} \, \mathrm{d}\mathsf{q}.$$

By taking the logarithm of both sides of the previous equation and by making use of Jensen's inequality [13], it can be shown that:

$$\log \mathbb{E}_{\mathsf{p}}\left[\exp\left(\rho \mathcal{J}(\mathbf{x}, t)\right)\right] \geq \int \rho \mathcal{J}(\mathbf{x}, t) \, \mathrm{d}\mathsf{q} - \mathbb{KL}\left(\mathsf{q} \| \mathsf{p}\right). \quad (5)$$

Let $\rho < 0$. By multiplying both sides of (5) with $-1/|\rho|$, one obtains:

$$\boxed{\xi(\mathbf{x}, t) = -\frac{1}{|\rho|} \mathcal{E}_{\mathsf{p}}\left(\mathcal{J}(\mathbf{x}, t); \rho\right) \leq \mathbb{E}_{\mathsf{q}}\left[\mathcal{J}(\mathbf{x}, t)\right] + \frac{1}{|\rho|} \mathbb{KL}\left(\mathsf{q} \| \mathsf{p}\right)} \qquad (6)$$

where $\mathbb{E}_{\mathsf{q}}\left[\mathcal{J}(\mathbf{x}, t)\right] = \int \mathcal{J}(\mathbf{x}, t) \, \mathrm{d}\mathsf{q}$. The inequality (6) provides us with a duality relationship between relative entropy and free energy. Essentially, one could define the following minimization problem:

$$-\frac{1}{|\rho|} \mathcal{E}_{\mathsf{p}}\left(\mathcal{J}(\mathbf{x}, t); \rho\right) = \inf_{\mathsf{q} \in \mathbf{P}(\Omega)} \left(\mathbb{E}_{\mathsf{q}}\left[\mathcal{J}(\mathbf{x}, t)\right] + \frac{1}{|\rho|} \mathbb{KL}(\mathsf{q} \| \mathsf{p})\right). \qquad (7)$$

It can be shown that the infimum in (7) is attained at q^*, where

$$\mathrm{d}\mathsf{q}^* = \frac{\exp\left(-|\rho| \mathcal{J}(\mathbf{x}, t)\right)}{\int \exp\left(-|\rho| \mathcal{J}(\mathbf{x}, t)\right) \mathrm{d}\mathsf{p}} \, \mathrm{d}\mathsf{p}. \qquad (8)$$

[1]Given two probability measures p and q, we say that q is *absolutely continuous* with p and write $\mathsf{q} \ll \mathsf{p}$ if $\mathsf{q} = 0 \Rightarrow \mathsf{p} = 0$, see page 161 of [12].

One can put q^* in (7) to verify that the right-hand side of the equation is indeed equal to its lower bound [13].

A rather intuitive way of writing (6) is to express it in the following form:

$$\underbrace{-\frac{1}{|\rho|} \mathcal{E}_{\mathsf{p}}\left(\mathcal{J}(\mathbf{x}, t); \rho\right)}_{\textbf{Helmholtz Free Energy}} \leq \underbrace{\text{Mean State Cost} + \frac{1}{|\rho|} \text{Information Cost}}_{\textbf{Non-Equilibrium Free Energy}} \qquad (9)$$

where "Mean State Cost" and "Information Cost" are defined as $\mathbb{E}_{\mathsf{q}}\left[\mathcal{J}(\mathbf{x}, t)\right]$ and $\mathbb{KL}\left(\mathsf{q} \| \mathsf{p}\right)$, respectively.

In the next sections, we derive the form of (7) for the case when $\mathbf{x}$ is the state of a nonlinear stochastic differential equation affine in noise and control.

A. Application of the Legendre Transformation to Stochastic Differential Equations

We consider the general uncontrolled and controlled stochastic dynamics affine in noise as follows:

$$\mathrm{d}\mathbf{x} = \mathbf{A}(\mathbf{x}) \, \mathrm{d}t + \mathbf{C}(\mathbf{x}) \, \mathrm{d}\mathbf{w}^{(0)}, \qquad (10)$$

$$\mathrm{d}\mathbf{x} = \mathbf{F}(\mathbf{x}, \mathbf{u}) \, \mathrm{d}t + \mathbf{C}(\mathbf{x}) \, \mathrm{d}\mathbf{w}^{(1)}, \qquad (11)$$

where $\mathbf{x} \in \mathbb{R}^n$ denotes the state of the system, $\mathbf{u} \in \mathbb{R}^m$ denotes the control input, $\mathbf{C}(\mathbf{x}) \in \mathbb{R}^{n \times m}$ is the diffusion matrix, $\mathbf{F}(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^n$ is the drift dynamics, and $\mathbf{w}^{(0),(1)} \in \mathbb{R}^m$ are Wiener processes (Brownian motion). The upper-scripts (0) and (1) are used to distinguish the two noise processes in the uncontrolled and controlled dynamics, respectively. The drift term $\mathbf{A}(\mathbf{x}) \in \mathbb{R}^n$ is defined by $\mathbf{A}(\mathbf{x}) = \mathbf{F}(\mathbf{x}, 0)$. The diffusion matrix may be partitioned as $\mathbf{C}(\mathbf{x}) = [\mathbf{0} \quad \mathbf{C}_c^{\mathsf{T}}(\mathbf{x})]^{\mathsf{T}}$ where $\mathbf{0} \in \mathbb{R}^{(n-m) \times m}$ and $\mathbf{C}_c(\mathbf{x}) \in \mathbb{R}^{m \times m}$ is invertible. Similarly, the drift term in the controlled dynamics may be partitioned as $\mathbf{F}(\mathbf{x}, \mathbf{u}) = [\mathbf{F}_1^{\mathsf{T}}(\mathbf{x}, \mathbf{u}) \quad \mathbf{F}_2^{\mathsf{T}}(\mathbf{x}, \mathbf{u})]^{\mathsf{T}}$ where $\mathbf{F}_1(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^{(n-m)}$ and $\mathbf{F}_2(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^m$; and the drift term in the uncontrolled dynamics may be partitioned as $\mathbf{A}(\mathbf{x}) = [\mathbf{A}_1^{\mathsf{T}}(\mathbf{x}) \quad \mathbf{A}_2^{\mathsf{T}}(\mathbf{x})]^{\mathsf{T}}$ where $\mathbf{A}_1(\mathbf{x}) \in \mathbb{R}^{(n-m)}$ and $\mathbf{A}_2(\mathbf{x}) \in \mathbb{R}^m$. The class of systems whose matrices can be partitioned as such contains rigid body, and multi body dynamics as well as kinematic models such as the ones considered in this work. Henceforth, for simplicity, we will assume that $m = n$. The case when $m < n$ can be treated similarly; see for instance [14]. Let $\mathbf{\Sigma}(\mathbf{x}) = \mathbf{C}(\mathbf{x}) \mathbf{C}^{\mathsf{T}}(\mathbf{x}) \in \mathbb{R}^{m \times m}$ and also define the following quantity:

$$\delta \mathbf{F}(\mathbf{x}, \mathbf{u}) = \mathbf{F}(\mathbf{x}, \mathbf{u}) - \mathbf{A}(\mathbf{x}) = \mathbf{F}(\mathbf{x}, \mathbf{u}) - \mathbf{F}(\mathbf{x}, 0), \quad \forall \mathbf{x}, \mathbf{u}.$$

To the system (11) we also associated the state cost

$$\mathcal{J}(\mathbf{x}(\cdot), t) = \Phi(\mathbf{x}(t_{\mathrm{f}})) + \int_t^{t_{\mathrm{f}}} q(\mathbf{x}(\tau), \tau) \, \mathrm{d}\tau. \qquad (12)$$

where the function $q : (x, t) \mapsto r$ returns a non-negative real number r for a given state x and time t. With a slight abuse of notation we will also use $\mathcal{J}(\mathbf{x}, t)$ to denote the value of $\mathcal{J}(\mathbf{x}(\cdot), t)$ along the trajectory $\mathbf{x}(\cdot)$ starting from $\mathbf{x} = \mathbf{x}(t)$ at time t. Expectations evaluated on trajectories generated by the uncontrolled dynamics and controlled dynamics will be represented by $\mathbb{E}_{\mathsf{p}}[\,\cdot\,]$ and $\mathbb{E}_{\mathsf{q}}[\,\cdot\,]$, respectively. The following fact can be found in [14].

Proposition 1: Given the measures p, q induced by the trajectories of (10) and (11), respectively, the *Radon-Nikodym derivative* of q with respect to p is defined by

$$\frac{\mathrm{d}\mathsf{q}}{\mathrm{d}\mathsf{p}} = \exp\left(\int_t^{t_f} \delta\mathbf{F}^\mathsf{T}(\mathbf{x}(\tau),\mathbf{u}(\tau))\mathbf{C}^{-1}(\mathbf{x}(\tau))\,\mathrm{d}\mathbf{w}^{(1)}(\tau)\right) + \exp\left(\int_t^{t_f} \tfrac{1}{2}\delta\mathbf{F}^\mathsf{T}(\mathbf{x}(\tau),\mathbf{u}(\tau))\boldsymbol{\Sigma}^{-1}(\mathbf{x}(\tau))\,\delta\mathbf{F}(\mathbf{x}(\tau),\mathbf{u}(\tau))\,\mathrm{d}\tau\right). \tag{13}$$

Given equation (13), the relative entropy term in (6) takes the form:

$$\frac{1}{|\rho|}\mathbb{KL}(\mathsf{q}\|\mathsf{p}) = \mathbb{E}_\mathsf{q}\left[\frac{1}{2|\rho|}\int_t^{t_f} \delta\mathbf{F}^\mathsf{T}(\mathbf{x}(\tau),\mathbf{u}(\tau))\boldsymbol{\Sigma}^{-1}(\mathbf{x}(\tau))\delta\mathbf{F}(\mathbf{x}(\tau),\mathbf{u}(\tau))\,\mathrm{d}\tau\right],$$

Substituting the previous expression of the Kullback-Leibler divergence into (6) one obtains

$$-\frac{1}{|\rho|}\mathcal{E}_\mathsf{p}\left(\mathcal{J}(\mathbf{x},t);\rho\right) \le \mathbb{E}_\mathsf{q}\left[\mathcal{J}(\mathbf{x},t)\right] + \mathbb{E}_\mathsf{q}\left[\frac{1}{2|\rho|}\int_t^{t_f} \delta\mathbf{F}^\mathsf{T}(\mathbf{x}(\tau),\mathbf{u}(\tau))\boldsymbol{\Sigma}^{-1}(\mathbf{x}(\tau))\delta\mathbf{F}(\mathbf{x}(\tau),\mathbf{u}(\tau))\,\mathrm{d}\tau\right].$$

The previous equation can be written in the form (9) with state cost term defined as $\mathbb{E}_\mathsf{q}\left[\mathcal{J}(\mathbf{x},t)\right]$ and information cost defined as $\mathbb{E}_\mathsf{q}\left[\frac{1}{2|\rho|}\int_t^{t_f} \delta\mathbf{F}^\mathsf{T}(\mathbf{x}(\tau),\mathbf{u}(\tau))\boldsymbol{\Sigma}^{-1}(\mathbf{x}(\tau))\delta\mathbf{F}(\mathbf{x}(\tau),\mathbf{u}(\tau))\,\mathrm{d}\tau\right]$. Next, we further specialize the class of systems where (9) is applied to, and discuss its connections to stochastic optimal control as in [1], [2], [13]. To this end, let us consider the special case of (10) and (11) with uncontrolled and controlled stochastic dynamics of the following form, respectively:

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x})\,\mathrm{d}t + \frac{1}{\sqrt{|\rho|}}\mathbf{B}(\mathbf{x})\,\mathrm{d}\mathbf{w}^{(0)}, \tag{14}$$

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x})\,\mathrm{d}t + \mathbf{B}(\mathbf{x})\left(\mathbf{u}\,\mathrm{d}t + \frac{1}{\sqrt{|\rho|}}\,\mathrm{d}\mathbf{w}^{(1)}\right), \tag{15}$$

where $\mathbf{x}\in\mathbb{R}^n$ denotes the state of the system, $\mathbf{B}(\mathbf{x})\in\mathbb{R}^{n\times m}$ is the control/diffusion matrix, $\mathbf{f}(\mathbf{x})\in\mathbb{R}^n$ is the passive dynamics, $\mathbf{u}\in\mathbb{R}^m$ is the control vector and $\mathbf{w}^{(0),(1)}$ are m-dimensional Wiener noise processes.

For the dynamics in (14) and (15) the form of the Radon-Nikodym derivative in (13) can be computed as follows. Noticing that $\delta\mathbf{F}(\mathbf{x},\mathbf{u}) = \mathbf{B}(\mathbf{x})\mathbf{u}$, $\mathbf{C}(\mathbf{x}) = \mathbf{B}(\mathbf{x})/\sqrt{|\rho|}$ and $\boldsymbol{\Sigma}(\mathbf{x}) = \mathbf{B}(\mathbf{x})\mathbf{B}^\mathsf{T}(\mathbf{x})/|\rho|$, and substituting these expressions in (13) yields

$$\frac{\mathrm{d}\mathsf{q}}{\mathrm{d}\mathsf{p}} = \exp\left(|\rho|\eta(\mathbf{u},t)\right) \quad \text{and} \quad \frac{\mathrm{d}\mathsf{p}}{\mathrm{d}\mathsf{q}} = \exp\left(-|\rho|\eta(\mathbf{u},t)\right), \tag{16}$$

where $\eta(\mathbf{u},t)$ is given by:

$$\eta(\mathbf{u},t) = \frac{1}{2}\int_t^{t_f}\mathbf{u}^\mathsf{T}(\tau)\mathbf{u}(\tau)\,\mathrm{d}\tau + \frac{1}{\sqrt{|\rho|}}\int_t^{t_f}\mathbf{u}^\mathsf{T}(\tau)\,\mathrm{d}\mathbf{w}^{(1)}(\tau). \tag{17}$$

Substitution of (16) and (17) into inequality (6) yields the following result:

$$-\frac{1}{|\rho|}\log\mathbb{E}_\mathsf{p}\left[\exp\left(-|\rho|\mathcal{J}(\mathbf{x},t)\right)\right] \le \mathbb{E}_\mathsf{q}\left[\mathcal{J}(\mathbf{x},t) + \eta(\mathbf{u},t)\right]. \tag{18}$$

Since the noise and the control terms are uncorrelated and the expectation of the noise is zero, the expectation on the right side of the inequality in (18) is further simplified as follows:

$$\underbrace{-\frac{1}{|\rho|}\log\mathbb{E}_\mathsf{p}\left[\exp\left(-|\rho|\mathcal{J}(\mathbf{x},t)\right)\right]}_{\xi(\mathbf{x},t)} \le \underbrace{\mathbb{E}_\mathsf{q}\left[\mathcal{J}(\mathbf{x},t) + \tfrac{1}{2}\int_t^{t_f}\mathbf{u}(\tau)^\mathsf{T}\mathbf{u}(\tau)\,\mathrm{d}\tau\right]}_{\text{Total Cost}}. \tag{19}$$

The right-hand side term in the above inequality corresponds to the cost function of a stochastic optimal control problem that is bounded from below by the free energy. Surprisingly, inequality (19) was derived without relying on any principle of optimality. Inequality (19) essentially defines a minimization process in which the right-hand side part of the inequality is minimized with respect to $\eta(\mathbf{u},t)$ and therefore with respect to the corresponding control $\mathbf{u}$. At the minimum, when $\mathbf{u}=\mathbf{u}^*$, the right-hand side of inequality in (19) attains its optimal value $\xi(\mathbf{x},t)$. Under the optimal control $\mathbf{u}^*$, and according to (8), the corresponding optimal distribution takes the form

$$\mathrm{d}\mathsf{q}^* = \frac{\exp\Big(-|\rho|\Phi(\mathbf{x}(t_\mathrm{f}))\Big)\exp\left(-|\rho|\int_t^{t_\mathrm{f}}q(\mathbf{x}(\tau),\tau)\,\mathrm{d}\tau\right)}{\int\exp\Big(-|\rho|\Phi(\mathbf{x}(t_\mathrm{f}))\Big)\exp\left(-|\rho|\int_t^{t_\mathrm{f}}q(\mathbf{x}(\tau),\tau)\,\mathrm{d}\tau\right)\mathrm{d}\mathsf{p}}\,\mathrm{d}\mathsf{p}. \tag{20}$$

The work [1], [2] inspired by early mathematical developments in control theory [13], [15], has shown that the value function $\xi(\mathbf{x},t)$ in (19) satisfies the Hamilton-Jacobi-Bellman equation and it has made the connection with more recent work in machine learning [16], [17] on Kullback-Leibler and path integral control.

B. Connection with Dynamic Programming (DP)

An important question that arises is: What is the link between (19) and the principle of optimality in dynamic programming? To address this question, we show that $\xi(\mathbf{x},t)$ satisfies the Hamilton-Jacobi-Bellman (HJB) equation associated with the optimal control problem (15)-(12) and hence, $\xi(\mathbf{x},t)$ is the corresponding value function of the following minimization problem

$$\begin{aligned}\xi(\mathbf{x},t) &= \\ &= \min_{\substack{\mathbf{u}(\tau)\\ t\le\tau\le t_\mathrm{f}}}\mathbb{E}_\mathsf{q}\left[\Phi(\mathbf{x}(t_\mathrm{f})) + \int_t^{t_\mathrm{f}}\left(q(\mathbf{x}(\tau),\tau) + \tfrac{1}{2}\mathbf{u}^\mathsf{T}(\tau)\mathbf{u}(\tau)\right)\mathrm{d}\tau\right] \\ &= \min_{\substack{\mathbf{u}(\tau)\\ t\le\tau\le t_\mathrm{f}}}\mathbb{E}_\mathsf{q}\left[\mathcal{J}(\mathbf{x},\tau) + \tfrac{1}{2}\int_t^{t_\mathrm{f}}\mathbf{u}^\mathsf{T}(\tau)\mathbf{u}(\tau)\,\mathrm{d}\tau\right],\end{aligned} \tag{21}$$

where the expectation is computed over the trajectories of (15). To see this, we introduce $\Psi(\mathbf{x},t) \triangleq \mathbb{E}_\mathsf{p}\left[\exp\left(\rho\mathcal{J}(\mathbf{x},t)\right)\right]$ and apply the Feynman-Kac lemma [18] to arrive at the backward Chapman-Kolmogorov partial differential equation (PDE)

$$\begin{aligned}-\partial_t\Psi(\mathbf{x},t) = &-|\rho|q(\mathbf{x},t)\Psi(\mathbf{x},t) + \mathbf{f}^\mathsf{T}(\mathbf{x})\nabla\Psi_\mathbf{x}(\mathbf{x},t) \\ &+ \frac{1}{2|\rho|}\operatorname{tr}\left(\nabla\Psi_{\mathbf{xx}}(\mathbf{x},t)\mathbf{B}(\mathbf{x})\mathbf{B}(\mathbf{x})^\mathsf{T}\right)\end{aligned} \tag{22}$$

with boundary condition $\Psi(\mathbf{x}(\mathrm{t_f}),\mathrm{t_f}) = \exp\big(-|\rho|\Phi(\mathbf{x}(\mathrm{t_f})\big)$, which governs the evolution of $\Psi(\mathbf{x},t)$ along the trajectories of (15) subject to $\mathbf{x} = \mathbf{x}(t)$. Since $\xi(\mathbf{x},t) = -\log\Psi(\mathbf{x},t)/|\rho|$, it follows that $\partial_t\Psi(\mathbf{x},t) = -|\rho|\Psi(\mathbf{x},t)\partial_t\xi(\mathbf{x},t), \nabla\Psi_{\mathbf{x}}(\mathbf{x},t) = -|\rho|\Psi(\mathbf{x},t)\nabla\xi_{\mathbf{x}}(\mathbf{x},t)$ and $\nabla\Psi_{\mathbf{xx}}(\mathbf{x},t) = |\rho|\Psi(\mathbf{x},t)\nabla\xi_{\mathbf{xx}}(\mathbf{x},t) - |\rho|^2\Psi(\mathbf{x},t)\nabla\xi_{\mathbf{x}}(\mathbf{x},t)\nabla\xi^{\mathsf{T}}_{\mathbf{x}}(\mathbf{x},t)$. In this case, it can be shown that $\xi(\mathbf{x},t)$ satisfies the nonlinear PDE

$$\begin{aligned}-\partial_t\xi(\mathbf{x},t) &= q(\mathbf{x},t) + \nabla\xi^{\mathsf{T}}_{\mathbf{x}}(\mathbf{x},t)\mathbf{f}(\mathbf{x})\\ &- \tfrac{1}{2}\nabla\xi^{\mathsf{T}}_{\mathbf{x}}(\mathbf{x},t)\mathbf{B}(\mathbf{x})\mathbf{B}^{\mathsf{T}}(\mathbf{x})\nabla\xi_{\mathbf{x}}(\mathbf{x},t)\\ &+ \frac{1}{2|\rho|}\operatorname{tr}\left(\nabla\xi_{\mathbf{xx}}(\mathbf{x},t)\mathbf{B}(\mathbf{x})\mathbf{B}^{\mathsf{T}}(\mathbf{x})\right),\end{aligned} \tag{23}$$

subject to the boundary condition $\xi(\mathbf{x}(\mathrm{t_f}),\mathrm{t_f}) = \Phi(\mathbf{x}(\mathrm{t_f}))$. The nonlinear PDE (23) corresponds to the HJB equation associated with the optimal control problem (21) and hence $\xi(\mathbf{x},t)$ is the corresponding minimizing value function [19]. It is important to note, however, that the principle of optimality was not used to derive (23).

C. Path Integral Control with Initial Sampling Policies

According to (19), one need to sample trajectories under the uncontrolled dynamics and evaluate the left-hand side of (19) on these trajectories in order to find the value function $\xi(\mathbf{x},t)$. However, in high-dimensional spaces, it is desirable to steer sampling toward specific areas of the state space. To do so, we have to incorporate an initial control policy into the uncontrolled dynamics. Therefore, instead of sampling from the uncontrolled dynamics (14), we sample trajectories based on the following stochastic dynamics:

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x})\,\mathrm{d}t + \mathbf{B}(\mathbf{x})\left(\mathbf{u}_{\mathrm{in}}\,\mathrm{d}t + \frac{1}{\sqrt{|\rho|}}\,\mathrm{d}\mathbf{w}^{(1)}\right), \tag{24}$$

where $\mathbf{u}_{\mathrm{in}}$ is an initial control policy. In [1], [2], the authors derived an iterative PI control without relying on previous policy parameterizations. More precisely, when sampling trajectories from the dynamics (24) the work in [1] and [2] showed that the value function $\xi(\mathbf{x},t)$ is expressed as

$$\xi(\mathbf{x},t) = -\frac{1}{|\rho|}\log\left(\int\exp\left(-|\rho|S\left(\mathbf{x},\mathbf{u}_{\mathrm{in}}(\mathbf{x},t),t\right)\right)\mathrm{dq}_{\mathrm{in}}\right)$$

where the term $S(\mathbf{x},\mathbf{u}_{\mathrm{in}})$ is defined as

$$\begin{aligned}S(\mathbf{x},\mathbf{u}_{\mathrm{in}}) = &\underbrace{\Phi(\mathbf{x}(\mathrm{t_f})) + \int_t^{t_\mathrm{f}} q(\mathbf{x}(\tau),\tau)\,\mathrm{d}\tau}_{\mathcal{J}(\mathbf{x},t)} +\\ &\underbrace{\frac{1}{2}\int_t^{t_f}\mathbf{u}^{\mathsf{T}}_{\mathrm{in}}(\tau)\mathbf{u}_{\mathrm{in}}(\tau)\,\mathrm{d}\tau + \frac{1}{\sqrt{|\rho|}}\int_t^{t_f}\mathbf{u}^{\mathsf{T}}_{\mathrm{in}}(\tau)\,\mathrm{d}\mathbf{w}^{(1)}(\tau)}_{\eta(\mathbf{u}_{\mathrm{in}},t)},\end{aligned} \tag{25}$$

where the term $\eta(\mathbf{u}_{\mathrm{in}},t)$ appears due to sampling based on the dynamics (24), while the term $\mathcal{J}(\mathbf{x},t)$ is the state-dependent part of the total cost function in (19). The path integral control is now expressed as [2]

$$\mathbf{u}_{\mathrm{PI}}(\mathbf{x},t)\,\mathrm{d}t = \mathbf{u}_{\mathrm{in}}(\mathbf{x},t)\,\mathrm{d}t + \delta\mathbf{u}(\mathbf{x},t), \tag{26}$$

where the term $\delta\mathbf{u}(\mathbf{x},t)$ is defined by

$$\delta\mathbf{u}(\mathbf{x},t) = \frac{1}{\sqrt{|\rho|}}\mathbb{E}_{\mathrm{q}^*}\left[\mathrm{d}\mathbf{w}^{(1)}\right] = \frac{1}{\sqrt{|\rho|}}\int\mathrm{d}\mathbf{w}^{(1)}\,\mathrm{dq}^*, \tag{27}$$

and where the expectation is taken under the optimal probability

$$\mathrm{dq}^* = \frac{\exp\left(-|\rho|S(\mathbf{x},\mathbf{u}_{\mathrm{in}})\right)}{\displaystyle\int\exp\left(-|\rho|S(\mathbf{x},\mathbf{u}_{\mathrm{in}})\right)\,\mathrm{dq}_{\mathrm{in}}}\,\mathrm{dq}_{\mathrm{in}}. \tag{28}$$

During implementation, equation (27) is approximated as

$$\begin{aligned}\delta\mathbf{u}(\mathbf{x},t) &= \frac{1}{\sqrt{|\rho|}}\sum_{k=1}^{\#\mathrm{traj}} p_k\mathrm{d}\mathbf{w}^{(1)}(\omega_k)\\ p_k &= \frac{\exp\left(-|\rho|S(\mathbf{x}_k,\mathbf{u}_{\mathrm{in}})\right)}{\sum_{\ell=1}^{\#\mathrm{traj}}\exp\left(-|\rho|S(\mathbf{x}_\ell,\mathbf{u}_{\mathrm{in}})\right)}\end{aligned} \tag{29}$$

The initial policy $\mathbf{u}_{\mathrm{in}}$ can be a suboptimal control law, a hand-tuned PD, PID control, or feedforward control. In this paper, we consider a feedforward control given by the RRT algorithm as the initial control policy. The RRT algorithm is proven to be a simple, iterative algorithm that quickly searches complicated, high-dimensional spaces for feasible paths. It grows a space-filling tree by drawing a random sample from the search space and connecting the nearest point in the tree to the new random sample at each iteration. This helps the tree to grow its branches toward unexplored regions of the search space quickly, i.e., achieving Voronoi bias [4]–[6]. In this case, the RRT-based optimal path integral control takes the form

$$\mathbf{u}_{\mathrm{PI}}(\mathbf{x},t)\,\mathrm{d}t = \mathbf{u}_{\mathrm{RRT}}(t)\,\mathrm{d}t + \delta\mathbf{u}(\mathbf{x},t). \tag{30}$$

In the next section, we discuss how to use the RRT algorithm to compute the initial control policy $\mathbf{u}_{\mathrm{RRT}}$.

IV. Trajectory Sampling via Sampling-based Algorithms

In high dimensional state spaces, sampling of useful trajectories from the unforced dynamics can be a tedious task. This issue can be addressed by first computing a "good enough" initial trajectory and then sampling local trajectories in the neighborhood of this trajectory. In the proposed approach, we use a probabilistic algorithm to find $\mathbf{u}_{\mathrm{in}}$ in (24) and compute an initial trajectory quickly. Probabilistic methods have proven to be very efficient for the solution of motion planning problems with dynamic constraints in high dimensional search spaces. Among them, Rapidly-exploring Random Trees (RRTs) [4]–[6] are among the most popular for solving single query motion planning problems. The main body of the RRT algorithm is given in Algorithm 1.

In the proposed approach, we leverage the speed and exploration capabilities of the RRT algorithm to compute an initial policy quickly by modifying the RRT primitive procedures. The proposed algorithm PI-RRT uses the path-integral approach to compute optimal trajectories that incorporate the system uncertainty (i.e., the risk of collision with obstacles). Since both final time and final state are given, the search space is formed by adding an additional time dimension T to the state space $\mathcal{X}$. Our search space, goal set and free space are thus defined as $\mathcal{Z} = \mathcal{X}\times T$, $\mathcal{Z}_{\mathrm{goal}} = \mathcal{X}_{\mathrm{goal}}\times T_{\mathrm{goal}}$, and $\mathcal{Z}_{\mathrm{free}} = \mathcal{Z}\setminus\mathcal{Z}_{\mathrm{goal}}$, respectively. The RRT algorithm is then run to find a trajectory starting from an initial point $z_{\mathrm{init}} = (x_{\mathrm{init}}, t_{\mathrm{init}})$ to the goal set $\mathcal{Z}_{\mathrm{goal}}$ while avoiding the obstacles in $\mathcal{X}$. The primitive procedures borrowed by the RRT algorithm are `Sample`, `Nearest`, `Steer` and Extend. In addition, given a trajectory $\boldsymbol{\sigma}$, the Boolean function ObstacleFree(σ) checks whether $\boldsymbol{\sigma}$ belongs to $\mathcal{Z}_{\text{free}}$ or not. It returns True if the trajectory is a subset of $\mathcal{Z}_{\text{free}}$, i.e., $\boldsymbol{\sigma} \subset \mathcal{Z}_{\text{free}}$, and False otherwise. Details for these procedures can be found in [20]. In addition, the PI-RRT algorithm uses the following procedures:

Steer : Given two points $\mathbf{z}_1$ and $\mathbf{z}_2$ in $\mathcal{Z}_{\text{free}}$, Steer extends $\mathbf{z}_1$ toward $\mathbf{z}_2$ by sampling trajectories from the unforced dynamics of the system. Specifically, the procedure samples a set of trajectories emanating from $\mathbf{z}_1$ and returns the closest end point of this set of trajectories to the point $\mathbf{z}_2$ with respect to a given distance function.

Extend: is a function that extends the nearest vertex of the graph $\mathcal{G}$ toward the randomly sampled point z_{rand}. Since time always flows in forward direction, we make sure that Extend computes valid connections, i.e., it returns false if the time value of z_{rand} is less than that of the nearest vertex in the graph. The Extend procedure of the RRT algorithm is shown in Algorithm 2.

ExtractPath: is a function that process on a tree data structure and extracts the information of the collision free trajectory starting from the current state to the goal set and the corresponding the control signal. The RRT Algorithm returns a tree $\mathcal{G}$ which contains the information of control inputs at each vertex of the tree at Line 5. Then, the ExtractPath procedure is subsequently called at the next line and it concatenates the control inputs by backtracking from the goal set toward the current state.

Execute: is a function that performs some initial portion of a given control signal on the system.

MeasureState: is a function that returns the current state of the system after it is executed with some control input. Since the system is subjected to noise, there is usually a difference between the real and the simulated state.

ComputeVariation: is a function that implements the path-integral control approach. After the RRT Algorithm computes a feasible trajectory and the corresponding control input, a fixed number of trajectories are locally sampled in the vicinity of this trajectory. Then, the correction term in control is computed as simply a weighted average of the noise profiles that create the local trajectories.

Algorithm 1: Body of the RRT Algorithm

```
1 RRT($\mathbf{z}_{\text{init}}$, $\mathcal{Z}_{\text{goal}}$, $\mathcal{Z}$)
2     $V \leftarrow \{\mathbf{z}_{\text{init}}\}$; $E \leftarrow \emptyset$;
3     $\mathcal{G} \leftarrow (V, E)$;
4     for $i = 1$ to $N$ do
5         $\mathbf{z}_{\text{rand}} \leftarrow$ Sample($i$);
6         $\mathcal{G} \leftarrow$ Extend($\mathcal{G}, \mathbf{z}_{\text{rand}}$);
7     return $\mathcal{G}$
```

The body of the path-integral based RRT algorithm is shown in Algorithm 3. It runs in a receding horizon fashion, that is, it computes a “good enough” control input and executes the first portion of the control signal at each time step. The algorithm starts by initializing the current time and state with the initial values in Lines 2-3. The algorithm then computes an initial policy in Line 5 by using the RRT algorithm. The steering procedure in the RRT algorithm is slightly modified in order to sample dynamically feasible trajectories. The steering procedure first samples a fixed number of trajectories from the unforced dynamics and then chooses the one that has the closest terminal state toward the desired point. Once a trajectory that reaches the goal set has been computed, the corresponding trajectory $\boldsymbol{\sigma}_{\text{RRT}}$, along with control the signal $\mathbf{u}_{\text{RRT}}$, are extracted from the computed data structure in Line 6. Then, the algorithm proceeds by locally sampling trajectories around $(\boldsymbol{\sigma}_{\text{RRT}}, \mathbf{u}_{\text{RRT}})$ and computes the variation in the control $\delta\mathbf{u}(\mathbf{x}, t)$ according to (27) by using information of local trajectories. Since we have M number of local trajectories, the expectation in (27) is numerically approximated by using the expression in (29). For each local trajectory $\boldsymbol{\sigma}_k$, a cost value is computed as $S(\boldsymbol{\sigma}_k, \mathbf{u}_{\text{RRT}})$ and its desirability value is computed by exponentiating the corresponding cost value, i.e., $d_k = \exp(-|\rho| S(\boldsymbol{\sigma}_k, \mathbf{u}_{\text{RRT}}))$. Then, the variation term in control $\delta\mathbf{u}(\mathbf{x}, t)$ is computed by taking the weighted average of all noise profiles which create the local trajectories and the weight of each trajectory is computed as the normalized desirability value, i.e., $p_k = d_k / \Sigma_{\ell=1}^{N} d_\ell$. The iteration of the algorithm is completed by executing the first τ times of the computed control signal and the algorithm keeps repeating the same steps until the final time is reached.

Algorithm 2: Extend Procedure for RRT Algorithm

```
1 Extend($\mathcal{G}$,$\mathbf{z}$)
2     $(V, E) \leftarrow \mathcal{G}$;
3     $\mathbf{z}_{\text{nearest}} \leftarrow$ Nearest($\mathcal{G}, \mathbf{z}$);
4     $(\mathbf{z}_{\text{new}}, \boldsymbol{\sigma}_{\text{new}}, \mathbf{u}_{\text{new}}) \leftarrow$ Steer($\mathbf{z}_{\text{nearest}}, \mathbf{z}$);
5     if ObstacleFree($\boldsymbol{\sigma}_{\text{new}}$) then
6         $V \leftarrow V \cup \{\mathbf{z}_{\text{new}}\}$;
7         $E \leftarrow E \cup \{(\mathbf{z}_{\text{nearest}}, \mathbf{z}_{\text{new}})\}$;
8     return $\mathcal{G}' \leftarrow (V, E)$
```

Algorithm 3: Body of the PI-RRT Algorithm

```
1 PI-RRT($\mathbf{z}_{\text{init}}$, $\mathcal{Z}_{\text{goal}}$, $\mathcal{Z}$)
2     $(t_{\text{i}}, \mathbf{x}_{\text{i}}) \leftarrow \mathbf{z}_{\text{init}}$; $(t_{\text{f}}, \mathcal{X}_{\text{goal}}) \leftarrow \mathcal{Z}_{\text{goal}}$;
3     $\mathbf{z}_{\text{i}} \leftarrow \mathbf{z}_{\text{init}}$;
4     while $t_{\text{i}} < t_{\text{f}}$ do
5         $\mathcal{G} \leftarrow$ RRT ($\mathbf{z}_{\text{i}}$, $\mathcal{Z}_{\text{goal}}$, $\mathcal{Z}$) ;
6         $(\boldsymbol{\sigma}_{\text{RRT}}, \mathbf{u}_{\text{RRT}}) \leftarrow$ ExtractPath($\mathcal{G}$);
7         $\delta\mathbf{u}_{[t_{\text{i}}, t_{\text{f}}]} \leftarrow$ ComputeVariation($\mathbf{u}_{\text{RRT}}, M$);
8         $\mathbf{u} \leftarrow \mathbf{u}_{\text{RRT}} + \delta\mathbf{u}$;
9         Execute($\mathbf{u}_{[t_{\text{i}}, t_{\text{i}}+\tau]}$);
10        $\mathbf{x}_{\text{i}} \leftarrow$ MeasureState($t_{\text{i}} + \tau$); $t_{\text{i}} \leftarrow t_{\text{i}} + \tau$;
11        $\mathbf{z}_{\text{i}} \leftarrow (\mathbf{x}_{\text{i}}, t_{\text{i}})$;
```

V. Comparison with Existing Methods

In [7], the authors introduced Chance Constrained Rapidly-exploring Random Trees (CC-RRT) algorithm to solve motion planning problems involving uncertainty in the location of the obstacles. Their approach is applicable for linear systems subject to process noise and/or uncertain obstacles which are assumed to be convex polyhedra. Due to the uncertainties in the problem, it may not be possible to identify a path guaranteed to be collision free surely. Therefore, the main idea in [7] is to relax this feasibility condition and introduce the notion of chance constraints, which guarantees probabilistic feasibility of computed trajectories. Under the assumption of

Gaussian noise, probabilistic feasibility at each time step can be established through simple simulation of the state conditional mean and the evaluation of linear constraints. After some algebraic operations, these probabilistic inequality constraints are converted into deterministic ones that yield conservative bounds in lieu of the inequality constraints representing the obstacles. Although both our approach and the CC-RRT algorithm leverage the RRT algorithm, they differ from each other in several ways. The PI-RRT algorithm can be applied to nonlinear systems which is affine in control and it does not impose any condition on the shape of the obstacles. Also, since the PI-RRT algorithm tries to compute the desirability function, i.e., the value function under exponentiation, it provides guarantees of optimality, whereas the CC-RRT algorithm only ensures path feasibility and relies on random sampling of controls to compute good enough trajectories without any guarantees.

VI. Numerical Simulations

In this section, we present a series of simulated experiments using a kinematic car model. We are interested in controlling a vehicle, whose motion is described by the following kinematic equations:

$$\dot{x} = v\cos\theta, \quad \dot{y} = v\sin\theta, \quad \dot{\theta} = w/r \tag{31}$$

where x, y are the Cartesian coordinates of a reference point of the vehicle, v is its speed, w is the control input and r is a positive constant. We assume that the admissible control inputs, are restricted by $w \in [-1, 1]$. We would like to find an optimal policy for the heading rate w to move the vehicle from a given initial configuration $(x_{\mathrm{i}}, y_{\mathrm{i}}, \theta_{\mathrm{i}})^{\mathsf{T}}$ to a final configuration $(x_{\mathrm{f}}, y_{\mathrm{f}}, \theta_{\mathrm{f}})^{\mathsf{T}}$ within some fixed final time t_{f}.

Let $\mathbf{x}_1 = x$, $\mathbf{x}_2 = y$, $\mathbf{x}_3 = \theta$ be the states and $\mathbf{u} = w$ be the control input of the system. Then (31) can be rewritten as

$$\dot{\mathbf{x}}_1 = v\cos\mathbf{x}_3, \quad \dot{\mathbf{x}}_2 = v\sin\mathbf{x}_3, \quad \dot{\mathbf{x}}_3 = \mathbf{u}/r. \tag{32}$$

Assuming the system is subjected to noise of intensity α in the control channel, (32) can be written in the standard form

$$\begin{pmatrix} \mathrm{d}\mathbf{x}_1 \\ \mathrm{d}\mathbf{x}_2 \\ \mathrm{d}\mathbf{x}_3 \end{pmatrix} = \begin{pmatrix} v\cos\mathbf{x}_3 \\ v\sin\mathbf{x}_3 \\ 0 \end{pmatrix} \mathrm{d}t + \begin{pmatrix} 0 \\ 0 \\ 1/r \end{pmatrix} (\mathbf{u}\,\mathrm{d}t + \alpha\,\mathrm{d}\mathbf{w}), \tag{33}$$

where $\mathbf{f}$, $\mathbf{B}$ and ρ in (24) are defined as follows

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} v\cos\mathbf{x}_3 \\ v\sin\mathbf{x}_3 \\ 0 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 1/r \end{pmatrix}, \quad \rho = -\frac{1}{\alpha^2}.$$

The following parameters were used in the numerical simulations: $\mathbf{x}_0 = (-9 \quad 0 \quad 0)^{\mathsf{T}}$, $t_0 = 0$, $\mathbf{x}_{\mathrm{f}} = (9 \quad 0 \quad 0)^{\mathsf{T}}$, $t_{\mathrm{f}} = 10$, $\mathrm{d}t = 0.1$, $v = 2.0$.

A. Example 1: Single-slit Obstacle

The objective in this problem is to find trajectories for the vehicle in a square environment with a box-like obstacle having a single slit. The trajectories computed by the PI-RRT algorithm at different stages are shown in Figure 1. The initial state is plotted as a yellow square and the goal region is shown in blue with magenta border (right-most). The computed path by the RRT algorithm following the unforced dynamics is shown in yellow. The locally sampled trajectories which are bundled around the yellow trajectory are shown in different colors. The trajectory of the vehicle due to execution of the control policy for some finite time horizon is shown in magenta.

To understand how the intensity of the noise level affects the patterns of the trajectories of the system, we run the algorithm and analyzed the situation for three different cases, $\alpha = 0.25$, 0.5 and 1.0 corresponding to low, medium and high intensity noise levels in the control channel. As shown in Figure 1 (a)-(c), the PI-RRT algorithm computes trajectories that pass through the slit most of the time when there is low intensity noise in the control channel. As a first step, the PI-RRT algorithm computes a baseline trajectory using the RRT algorithm. The vertices and the edges of the tree computed by the RRT algorithm are shown in green and blue colors, respectively. During the simulations, it was observed that this baseline trajectory does not necessarily pass through the slit. The RRT algorithm sometimes returns a baseline trajectory that passes close by the upper or the lower sections of the obstacle due to both the noise which is observed in the dynamics and the randomized nature of the algorithm itself. The PI-RRT algorithm then samples a bundle of trajectories around the baseline trajectory in order to compute the variation term for the new control input. The new control input is computed by summing up the baseline control policy returned by the RRT algorithm and the variation term, which is the weighted average of the contribution of each locally sampled trajectory. These weights are computed by using the cost information of each locally sampled trajectory. We observed that the distribution of the trajectories, which pass close to the upper or lower corners or through the slit, changes as the intensity of the noise increases. For higher intensity of the noise, the PI-RRT algorithm computes trajectories which do not pass through the slit but rather pass close to the upper or lower corners. This change in the distribution of trajectories is shown in Figures 1 (d)-(f) for medium intensity noise and in Figures 1 (g)-(i) for high intensity noise.

B. Example 2: Double-slit Obstacle

Next, we consider a more challenging motion planning problem. In this case, there are two slits on the obstacle block and the length of the slits is longer than in the previous example. The longer length of the slits results in a higher probability of collision while traversing through the slit, which makes the motion planning problem more challenging.

A study was performed in order to compare the performance of the PI-RRT algorithm with the RRT algorithm. No variation term in the control input was computed for the RRT algorithm, and it was simply executed in a receding horizon fashion. All algorithms were run for 6000 iterations to find a baseline trajectory. The results over 100 trials are shown in Figures 2, 3 and 4. The trajectories that result in collision are plotted in Figure 2 (a), (d) for the low noise level, Figure 3 (a), (d) for the medium noise level, and Figure 4 (a), (d) for the high noise level for the RRT and PI-RRT algorithms, respectively. Also, the distribution of collision-free trajectories is plotted in Figure 2 (c), (f) for the low noise level, Figure 3 (c), (f) for the medium noise level, and Figure 4 (c), (f) for the high noise level for the RRT and PI-RRT algorithms,

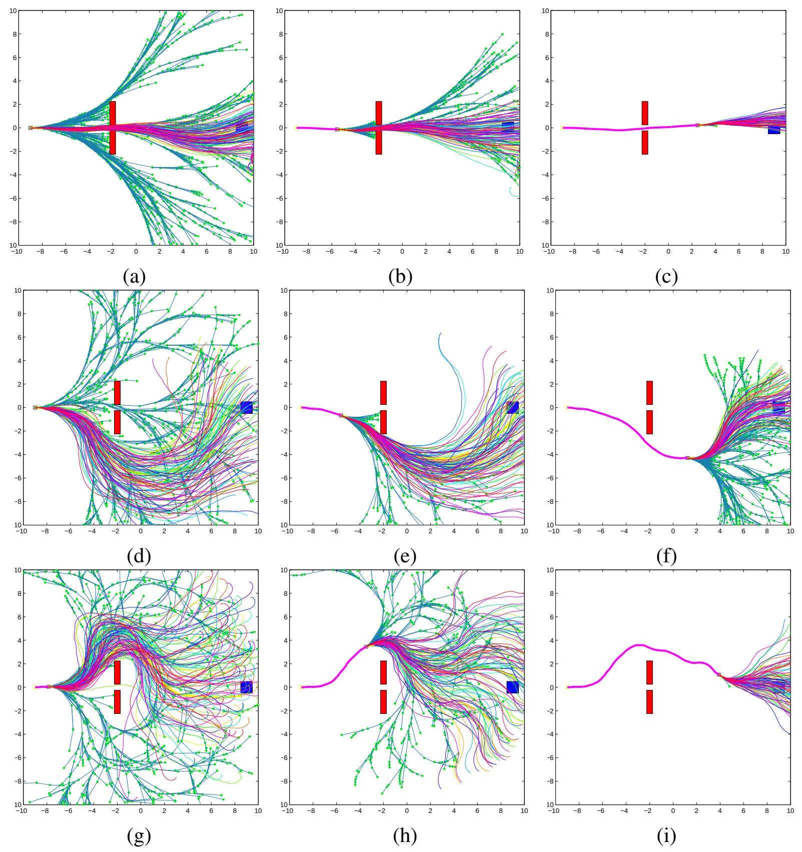
Figure 1. The trajectories computed by the PI-RRT algorithm for stochastic optimal control of the kinematic car model under different levels of noise injected to the control channel: (a)-(c) is with $\alpha = 0.25$, (d)-(f) is with $\alpha = 0.50$, and (g)-(i) is with $\alpha = 1.0$.

respectively. The distribution of trajectories and the number of trajectories which result in a collision are summarized in Table I. Under the 'Success' column, the rows of the table contain the number of collision-free trajectories which pass through the bottom corner, bottom slit, top slit and top corner of the block. As shown in Table I, the PI-RRT computes safer control policies which reduce the risk of having a collision. On the other hand, both the RRT and the PI-RRT compute trajectories that are almost equally distributed over both slits.

In summary, it was observed that the behaviors of both algorithms are similar for the case with high noise level. As the noise level decreases, most of the failed cases, not surprisingly, occur when the algorithms try to compute a path that passes through the slits. Our simulation results demonstrate that the PI-RRT algorithm tends to compute trajectories that have larger clearance from obstacles and hence outperforms the standard RRT algorithm, resulting in a smaller failure rate.

Table I. MONTE-CARLO RESULTS FOR DOUBLE-SLIT OBSTACLE

	$\alpha = 0.25$					$\alpha = 0.50$					$\alpha = 1.00$				
	Success				Fail	Success				Fail	Success				Fail
RRT	0	24	20	0	56	23	8	11	27	31	48	0	0	44	8
PI-RRT	0	44	45	0	11	35	9	8	37	11	47	0	0	49	4

VII. CONCLUSION

In this paper, the PI-RRT algorithm is proposed in order to solve a class of stochastic optimal control problems. The proposed approach makes a novel connection between incremental sampling-based algorithms and path integral control. The work in this paper can be extended in several directions. First, a parallel version of the algorithm can be implemented by sampling local trajectories or computing several initial trajectories simultaneously. Second, since there exist many variants of the standard RRT algorithm, one can implement different sampling-based algorithms to compute initial trajectories and incorporate them within the path integral framework. For example, the RRT^* [20] and the $\mathrm{RRT}^\#$ algorithms [21], which are both asymptotically optimal, can be used to compute bundles of good initial trajectories in a single pass; however, such an algorithm would require more elaborate computations for implementing the steering function, e.g., backward integration of a stochastic differential equation. This is part of ongoing work. Finally, we plan to apply PI-RRT to robotic systems with many states/degrees of freedom and compare it with CC-RRT.

REFERENCES

[1] E.A. Theodorou and E. Todorov. Relative entropy and free energy dualities: Connections to path integral and kl control. In *the Proceedings of IEEE Conference on Decision and Control*, pages 1466–1473, Dec 2012.

[2] E. Theodorou, D. Krishnamurthy, and E. Todorov. From information theoretic dualities to path integral and kullback-leibler control: Continuous and discrete time formulations. In *The Sixteenth Yale Workshop on Adaptive and Learning Systems*.

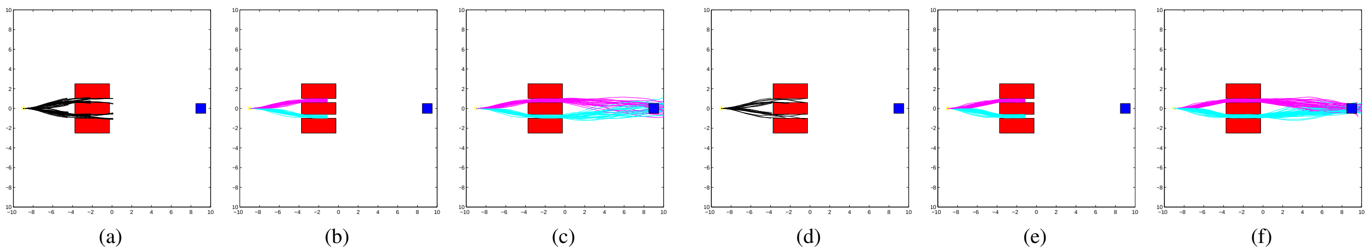

Figure 2. Distribution of trajectories for kinematic car model under low intensity of noise injected to the control channel ($\alpha = 0.25$) is shown in (a)-(c) for the RRT algorithm, and in (d)-(f) for the PI-RRT algorithm. The trajectories which hit the obstacles are shown in (a), (d). The collision-free trajectories at an intermediate stage are shown in (b), (e), and at the final stage are shown in (c), (f).

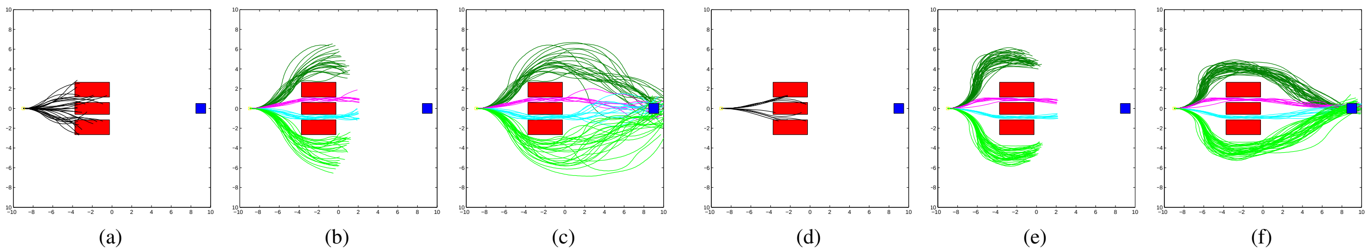

Figure 3. Distribution of trajectories for kinematic car model under medium intensity of noise injected to the control channel ($\alpha = 0.50$) is shown in (a)-(c) for the RRT algorithm, and in (d)-(f) for the PI-RRT algorithm. The trajectories which hit the obstacles are shown in (a), (d). The collision-free trajectories at an intermediate stage are shown in (b), (e), and at the final stage are shown in (c), (f).

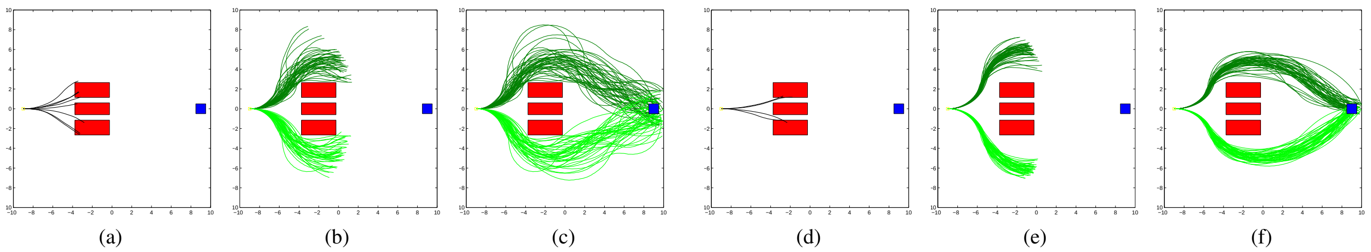

Figure 4. Distribution of trajectories for kinematic car model under high intensity of noise injected to the control channel ($\alpha = 1.0$) is shown in (a)-(c) for the RRT algorithm, and in (d)-(f) for the PI-RRT algorithm. The trajectories which hit the obstacles are shown in (a), (d). The collision-free trajectories at an intermediate stage are shown in (b), (e), and at the final stage are shown in (c), (f).

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

[4] S. M. LaValle and J. J. Kuffner, Jr. Rapidly-exploring random trees: Progress and prospects. In B. R. Donald, K. Lynch, and D. Rus, editors, *New Directions in Algorithmic and Computational Robotics*, pages 293–308. 2001.

[5] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[6] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Intelligent Robotics and Autonomous Agents. The MIT Press, May 2005.

[7] B. Luders, M. Kothari, and J. P. How. Chance constrained RRT for probabilistic robustness to environmental uncertainty. In *AIAA Guidance, Navigation, and Control (GNC)*, Toronto, Canada.

[8] J. Buchli, F. Stulp, E. Theodorou, and S. Schaal. Learning variable impedance control. *The International Journal of Robotics Research*, 30(7):820–833, April 2011.

[9] F. Stulp, E.A. Theodorou, and S. Schaal. Reinforcement learning with sequences of motion primitives for robust manipulation. *IEEE Transactions on Robotics*, 28(6):1360–1370, 2012.

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

[11] Jan Peters, Katharina Mülling, and Yasemin Altun. Relative entropy policy search. In *Proceedings of the Twenty-Fourth National Conference on Articial Intelligence*, 2010.

[12] B.K. Øksendal. *Stochastic differential equations : An introduction with Applications*. Springer, Berlin ; New York, 6th edition, 2003.

[13] P. Dai Pra, L. Meneghini, and W. Runggaldier. Connections between stochastic control and dynamic games. *Mathematics of Control, Signals, and Systems*, 9(4):303–326, December 1996.

[14] J. Yang and J. H. Kushner. A Monte Carlo method for sensitivity analysis and parametric optimization of nonlinear stochastic systems. *SIAM Journal in Control and Optimization*, 29(5):1216–1249, 1991.

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

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

[17] E. Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences*, 106(28):11478–11483, 2009.

[18] A. Friedman. *Stochastic Differential Equations and Applications*. Dover Books on Mathematics. Dover Publications, December 2006.

[19] R. F. Stengel. *Optimal Control and Estimation*. Dover Publications, New York, 1994.

[20] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[21] O. Arslan and P. Tsiotras. Use of relaxation methods in sampling-based algorithms for optimal motion planning. In *IEEE International*

Conference on Robotics and Automation, pages 2421–2428, Karlsrühe, Germany, May 6–10, 2013.