Advancing Frontiers of Path Integral Theory for Stochastic Optimal Control

by

Apurva Patil

Ph.D. Proposal

Submitted to the Department of Mechanical Engineering

The University of Texas at Austin

In Partial Fulfillment of the Requirements

For Ph.D. Candidacy in Mechanical Engineering

November 2024

The University of Texas at Austin

Abstract

Stochastic Optimal Control (SOC) problems arise in systems where the dynamics are influenced by random disturbances and uncertainties, such as robotic systems navigating unpredictable environments or financial systems subject to asset price fluctuations. These problems are modeled using stochastic differential equations (SDEs), where the goal is to design control inputs that minimize a cost function while accounting for the probabilistic nature of the system's evolution. Traditional methods like dynamic programming, though powerful, are computationally expensive, particularly for high-dimensional and non-linear systems due to the *curse of dimensionality*. The path integral control approach offers a promising alternative for solving SOC problems, especially in complex, high-dimensional systems. Based on the Feynman-Kac theorem, the path integral method transforms the SOC problem into an expectation over noisy system trajectories, enabling efficient policy computation through Monte Carlo sampling. As the number of Monte Carlo samples approaches infinity, the policy generated by the path integral method converges to an optimal policy. Due to its purely simulator-driven nature, path integral control is applicable to high-dimensional, nonlinear, and SOC problems, where conventional model-based policy synthesis methods often face challenges. The ability to parallelize the Monte Carlo simulations on Graphics Processing Units (GPUs) makes the path integral approach scalable and well-suited for real-time applications. Additionally, it operates directly on the system's non-linear dynamics, allowing it to handle a wide range of practical systems where traditional linear approximations fall short. The path integral approach is inherently robust to uncertainties, making it particularly relevant for safety-critical applications such as autonomous vehicle navigation, where accounting for sensor noise and unpredictable conditions is essential for safe operation. Its real-time policy generation capability makes it adaptable to dynamic environments, further enhancing its utility in autonomous systems and robotics. This dissertation develops and applies the path integral control theory to six classes of SOC problems: Chance-Constrained SOC, Two-Player Zero-Sum Stochastic Differential Games, Deceptive Control, Task Hierarchical Control, Risk Mitigation of Stealthy Attack and Discrete-Time Stochastic Linear Quadratic Regulator (LQR). Additionally, it presents a sample complexity analysis of the path integral controller for discrete-time stochastic LQR problems. Through these contributions, this research work lays foundational groundwork for simulator-driven autonomy in solving complex SOC problems, particularly those involving non-linear dynamics and high-dimensional state spaces.

Contents

1 Introduction

Making the best possible decisions within the available time and computational resource constraints is a necessary skill for truly autonomous systems. In many engineering scenarios, optimal decision-making requires solving Stochastic Optimal Control (SOC) problems. SOC refers to a class of control problems that involves making optimal decisions for systems whose dynamics are influenced by random disturbances or uncertainties. These uncertainties can arise from various factors such as environmental noise, model inaccuracies, or unobserved phenomena. Examples include robotic systems navigating uncertain environments, financial systems where asset prices evolve stochastically, and energy systems subject to demand fluctuations. Unlike deterministic control problems, where the system's behavior can be predicted exactly, SOC problems account for the randomness in the system. The general SOC problem involves a dynamical system described by stochastic differential equations (SDEs). These equations model the evolution of the system's state over time, driven by both control inputs and stochastic processes, typically represented as *Wiener* or *Brownian motion* [1]. The challenge lies in designing control inputs that minimize a given cost function, such as energy consumption or deviation from a target state, while considering the probabilistic evolution of the system. This cost function is often an expectation over all possible system trajectories, making the optimization inherently complex.

At the core of SOC is the Bellman principle of optimality, which states that the optimal policy at any given time depends only on the current state of the system, not on the prior history of states. This leads to the formulation of the Hamilton-Jacobi-Bellman (HJB) equation, a partial differential equation (PDE) that characterizes the value function of the SOC problem. Solving the HJB PDE yields the optimal control policy, but for high-dimensional systems and non-linear dynamics, solving the HJB equation analytically is often intractable. One of the commonly used approach to solve HJB PDE is *dynamic programming* which breaks down the SOC problem into smaller subproblems by recursively solving the HJB PDE. While powerful, this method suffers from the *curse of dimensionality*, where the computational cost grows exponentially with the number of state variables. This makes the traditional dynamic programming approaches impractical for real-time control.

The path integral approach, rooted in the principles of statistical physics, offers an alternative method for solving nonlinear SOC problems online [2, 3, 4]. The fundamental insight behind path integral control lies in the Feynman-Kac theorem [5], a result from stochastic calculus that connects PDEs with stochastic processes. This theorem allows the transformation of the HJB equation into an expectation of an exponential cost function over noisy system trajectories which allows for efficient computation of optimal policies through Monte Carlo sampling methods. According to the strong law of large numbers [1], as the number of Monte Carlo samples approaches infinity, the policy generated by the path integral method converges to an optimal policy. The path integral method can be flexibly applied to high-dimensional control problems that involve nonlinear, stochastic, and hybrid dynamics – areas where conventional model-based approaches often face challenges. Moreover, it relies on Monte Carlo simulations that can be massively parallelized on Graphics Processing Units (GPUs). Millions of trajectories can be simulated simultaneously significantly reducing computation time making the path integral approach less susceptible to the curse of dimensionality [6]. Moreover, in certain cases, it can be formally shown that the required number of samples of Monte Carlo simulations grows only logarithmically as a function of the dimension of the control input [7], in contrast to exact dynamic programming, whose computational cost grows exponentially.

The path integral approach is inherently robust to uncertainties. By accounting for the stochastic nature of the system, it ensures that control policies are not only optimized for expected outcomes but also for handling variability in system behavior. This makes the path inetgral method highly relevant for safety-critical applications, where uncertainties can lead to catastrophic failures if not properly managed. For instance, in autonomous vehicle navigation, the ability to optimize trajectories under uncertain road conditions or sensor noise is crucial for ensuring safe operation. Another significant benefit of the path integral control approach is its ability to handle non-linear systems. Many real-world systems, such as autonomous robots or drones, have non-linear dynamics that make traditional control methods, which rely on linear approximations, inadequate. The path integral method, however, operates directly on the system's non-linear dynamics, making it well-suited for controlling systems where non-linearities are inherent and crucial to their behavior. Moreover, the path integral approach is particularly effective for online computation of control policies which is important for applications such as autonomous driving or robotic navigation, where decisions must be made in real time. The path integral method can generate control inputs dynamically as new information becomes available, making it adaptable to rapidly changing environments. Finally, the path integral approach determines control inputs solely through simulators, bypassing the need for analytical models of the plant. This is particularly beneficial in scenarios where a simulator engine exists, but creating a mathematical (equation-based) model of the system poses challenges. As applications in robotics, autonomous systems, and other fields continue to grow, the path integral control approach is likely to play a critical role in advancing the state of the art in control theory and practice.

1.1 History of Path Integral Control

The Path Integral Control is a control algorithm inspired by the Path Integral formulation of quantum mechanics. The origin of path integral control can be traced back to the stochastic variational treatment of quantum mechanics [8, 9, 10, 11, 12, 13]. Using Nelson's diffusion model [8], references [12, 13] derived an SOC problem whose associated HJB equation coincides with the linear Schrödinger equation. This connection between SOC and quantum mechanics was exploited by Itami [14, 15], who proposed a Monte Carlo (Metropolis) algorithm to evaluate the path integral representation of the wave function. Noticing the semi-classical limit of the Schrödinger equation approximates the HJB equation, [15] proposed a Monte Carlo-based solution approach to deterministic nonlinear optimal control problems. This is the first appearance of what we now know as path integral control to the best of the our knowledge. However, it should be noted that mathematically relevant problems, such as risk-sensitive control [16, Ch. VI],[17, 18], distributionally robust control against relative entropy ambiguity set [19], and linearly solvable Markov-Decision Processes (MDP) [20, 21] (also known as the KL control problems) have been investigated in separate research threads.The line of work [20, 21] has been broadly accepted by the neuroscience community, where linearly solvable MDPs are now widely used to reason about brain's functionality in spatial navigation (e.g., [22]).

Kappen [2] showed that a certain class of *stochastic* optimal control problems can be solved by the path integral method without semi-classical approximation. For such SOC problems, it was shown that the associated stochastic HJB equation can be linearized by the Cole-Hopf transformation (a typical change-of-variables technique to relate the HJB equation with Burgers' equation and the heat equation [23]). The Feynman-Kac lemma [5] is then invoked to show that the solution to the linearized PDE admits a path integral representation, which can be numerically evaluated by Monte Carlo simulations. The authors of [3, 24] demonstrated the path integral for policy improvements (PI^2), who pioneered significant developments of path integral control in robot learning [25, 26, 27, 28, 29, 30]. A receding horizon implementation of path integral control, called Model Predictive Path Integral (MPPI) control [31, 6], demonstrated aggressive driving [4], where real-time Monte Carlo simulations were parallelized on GPUs. The path integral control method has also been applied to constrained systems and systems with non-differentiable dynamics [32, 33].

An alternative derivation of path integral control bypassing the Feynman-Kac lemma was presented in the work [34]. In [34], the authors used the variational lower bound and the Girsanov theorem to establish a connection between the linearly solvable MDP (the KL control problems) and the problem studied by Kappen [2]. Yet another derivation of path integral control is possible by formulating an SOC as a statistical inference problem, and then applying the standard graphical model inference algorithms [35, 36].

It is worth noting that, in all derivations, the path integral control method faces the same limitation. It is optimal only for the class of SOC problems whose dynamic programming equations (e.g., HJB PDEs) are linearizable. Removal of this restriction has been studied in the literature. In [37], the authors proposed an iterative applications of the Feynman-Kac lemma. The MPPI framework [31] utilizes a heuristic approach based on the KL-divergence minimization. However, this limitation has not been satisfactorily removed as of today.

1.2 Organization of the Proposal

In this work, we develop path integral theory to solve six classes of stochastic optimal control problems as depicted in Figure 1. This dissertation proposal is organized as follows: we develop the path integral control theory for six classes of

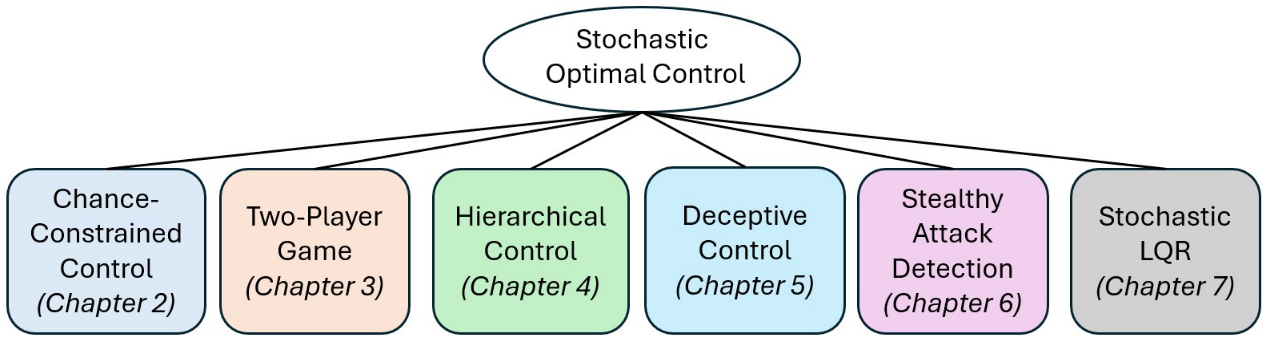

Stochastic Optimal Control
Chance-Constrained Control (Chapter 2)
Two-Player Game (Chapter 3)
Hierarchical Control (Chapter 4)
Deceptive Control (Chapter 5)
Stealthy Attack Detection (Chapter 6)
Stochastic LQR (Chapter 7)

Figure 1: Dissertation Proposal Structure

stochastic optimal control problems: Chance-Constrained SOC (Chapter 2), Two-Player Zero-Sum Stochastic Differential

Games (Chapter 3), Deceptive Control (Chapter 5), Task Hierarchical Control (Chapter 4), Detection and Risk Mitigation of Stealthy Attack (Chapter 6) and Stochastic LQR Problem (Chapter 7). Chapter 7 also derives the sample complexity of the path integral controller applied to a discrete-time stochastic LQR problem. Chapter 8 details the proposed dissertation structure and timeline. Chapter 9 provides a list of publications resulting from this work, and finally, Chapter 11 is dedicated to the appendix.

2 Chance-Constrained Stochastic Optimal Control

2.1 Motivation

In safety-critical missions, quantitative characterization of system uncertainties is of critical importance as it impacts the overall safety of the system operation. System uncertainties arise due to unmodeled dynamics, unknown system parameters, and external disturbances. Subsequently, the control policies should be designed to accommodate such uncertainties in order to achieve user-defined safety requirements. Robust control is a popular paradigm to guarantee safety against set-valued uncertainties [38, 39]. Generally, a robust control approach aims to synthesize a policy that optimizes the worst-case performance, commonly known as the minimax policy. While such a strategy is suitable for applications where safety is an absolute requirement, the computation of the exact minimax policy is often intractable, which necessitates a sequential outer-approximation of the uncertain sets. This often results in overly conservative solutions [40]. Also, robust control is difficult to apply when the uncertainty is modeled probabilistically using random variables with unbounded support (e.g., Gaussian distributions).

Chance-constrained stochastic optimal control (SOC) is an alternative paradigm for policy synthesis under uncertainty [41, 42]. Unlike robust control, this approach aims to optimize the system performance by accepting a user-specified threshold for the probability of failure (e.g., collisions with obstacles). Notably, the acceptance of the possibility of failure is often effective to reduce the conservatism of the controller even if the introduced probability of failure is practically negligible [43]. Consequently, chance-constrained SOC is also widely used as a framework for policy synthesis for systems in safety-critical missions.

2.2 Literature Review

In this work, we consider a continuous-time continuous-space chance-constrained SOC problem. A central challenge in this problem stems from the fact that the continuous-time end-to-end probability of failure is generally challenging to evaluate and optimize against. A common course of action found in the literature is to look for a tractable approximation of the chance-constrained SOC problem in order to use existing tools from optimization. In [41] a discrete-time chance-constrained SOC problem is converted to a disjunctive convex program by approximating the chance constraint using Boole's inequality. The disjunctive convex program is then solved using branch-and-bound techniques. The work presented in [44] solves the discrete-time chance-constrained problem by formulating its dual. However, in this work, the joint chance constraint is conservatively approximated using Boole's inequality. Hence, the duality gap is nonzero and the obtained solution is suboptimal. In [45], authors use statistical moments of the distributions in concentration inequalities to upper-bound the chance constraint for discrete-time trajectory planning under non-Gaussian uncertainties, and solve the problem using nonlinear program solvers. A scenario-based optimization method that translates the chance constraints into deterministic ones is provided in [46]. Taking ideas from distributionally robust optimization, deterministic convex reformulation of the chance-constrained stochastic optimal control problems is proposed in [47] for discrete-time linear system dynamics. Different from the above works in our work, we consider continuous-time, control-affine system dynamics. For the continuous-time chance-constrained planning problem the authors in [48] use risk contours to transform the original problem into a deterministic planning problem and use convex methods based on sum-of-squares optimization to obtain continuous-time trajectories. In [49] a continuous-time chance-constrained SOC problem is converted to a deterministic control problem with convex constraints using generalized polynomial chaos expansion and Chebyshev inequality. The deterministic optimal control problem is then solved using sequential convex programming. Other approaches that consider approximations of the chance constraints include approaches based on concentration of measure inequalities [50], Bernstein approximation [51], and moment based surrogate [52]. A common limitation of the above approaches is the conservatism introduced by the approximation of the chance constraints, leading to overly cautious policies that compromise performance or cause artificial infeasibilities [51, 53, 54, 55]. Deep reinforcement learning algorithms such as soft actor-critic also have been used to solve the chance-constrained SOC problems [56].

2.3 Contributions

In our work, we utilize the path integral approach to numerically solve the posed chance-constrained SOC problem using open-loop samples of system trajectories. The contributions of this work are as follows:

1. We leverage the notion of exit time from the continuous-time stochastic calculus [57, 58] to formulate a chance-constrained SOC problem. The dual of this chance-constrained SOC problem is constructed by incorporating the chance constraint into the cost function. The chance constraint is then transformed into an expectation of an indicator function, which enjoys the same additive structure as the primal cost function. No approximation nor time discretization is introduced in this transformation.

2. Given a fixed dual variable, we evaluate the dual objective function by numerically solving the Hamilton-Jacobi-Bellman (HJB) partial differential equation (PDE).

3. It is shown that under a certain assumption on the system dynamics and cost function, a strong duality holds between the primal problem and the dual problem (the duality gap is zero).

4. We propose a novel path-integral-based dual ascent algorithm to numerically solve the dual problem. This allows us to solve the original chance-constrained problem online via open-loop samples of system trajectories. Finally, we present simulation studies on chance-constrained motion planning for spatial navigation of mobile robots. The solution obtained using the path integral approach is compared with that of the finite difference method.

Notations

Let $\mathbb{R}$, $\mathbb{R}^n$, and $\mathbb{R}^{m\times n}$ be the set of real numbers, the set of n-dimensional real vectors, and the set of $m \times n$ real matrices. If a stochastic process $x(t)$ starts from x at time t, then let $P_{x,t}(\mathcal{E}) = P\big(\mathcal{E}(x) \mid x(t) = x\big)$ denote the probability of event $\mathcal{E}(x)$ involving the stochastic process $x(t)$ conditioned on $x(t) = x$, and let $\mathbb{E}_{x,t}[F(x)] = \mathbb{E}[F(x) \mid x(t) = x]$ denote the expectation of $F(x)$ (a functional of $x(t)$) also conditioned on $x(t) = x$. Let $\mathbb{1}_{\mathcal{E}}$ be an indicator function, which returns 1 when the condition $\mathcal{E}$ holds and 0 otherwise. The $\bigvee$ symbol represents a logical OR implying existence of a satisfying event among a collection. Tr(A) denotes the trace of a matrix A. ∂_x and ∂_x^2 are used to define, respectively, the first and second-order partial derivatives w.r.t. x. If x is a vector then ∂_x returns a column vector and ∂_x^2 returns a matrix. The short forms a.a. and a.s. denote almost all and almost surely, respectively.

2.4 Preliminaries

Let $\mathcal{X}_s \subseteq \mathbb{R}^n$ be a bounded open set representing a safe region, $\partial\mathcal{X}_s$ be its boundary, and closure $\overline{\mathcal{X}_s} = \mathcal{X}_s \cup \partial\mathcal{X}_s$.

2.4.1 Controlled and Uncontrolled Processes

Consider a controlled process $x(t) \in \mathbb{R}^n$ driven by following control-affine Itô stochastic differential equation (SDE):

$$dx(t) = f(x(t), t)\, dt + G(x(t), t)\, u(x(t), t) dt + \Sigma(x(t), t)\, dw(t), \tag{1}$$

where $u(x(t), t) \in \mathbb{R}^m$ is a control input, $w(t) \in \mathbb{R}^k$ is a k-dimensional standard Wiener process on a suitable probability space $(\Omega, \mathcal{F}, P)$, $f(x(t), t) \in \mathbb{R}^n$, $G(x(t), t) \in \mathbb{R}^{n\times m}$ and $\Sigma(x(t), t) \in \mathbb{R}^{n\times k}$. Let $\hat{x}(t) \in \mathbb{R}^n$ be an uncontrolled process driven by the following SDE:

$$d\hat{x}(t) = f(\hat{x}(t), t) dt + \Sigma(\hat{x}(t), t) dw(t). \tag{2}$$

At the initial time t_0, $x(t_0) = \hat{x}(t_0) = x_0 \in \overline{\mathcal{X}_s}$. Throughout this work, we assume sufficient regularity in the coefficients of (1) and (2) so that unique strong solutions exist [59, Chapter 1]. In the rest of the dissertation, for notational compactness, the functional dependencies on x and t are dropped whenever it is unambiguous.

2.4.2 Probability of Failure

For a given finite time horizon $t \in [t_0, T]$, $t_0 < T$, if the system (1) leaves the safe region $\mathcal{X}_s$ at any time $t \in (t_0, T]$, then we say that it fails.

Definition 1 (Probability of failure)**.** *The probability of failure* $P_{\text{fail}}\left(x_0, t_0, u(\cdot)\right)$ *of system (1) starting at* (x_0, t_0) *and operating under the policy* $u(\cdot)$ *is defined as*

$$P_{\text{fail}}\left(x_0, t_0, u(\cdot)\right) = P_{x_0,t_0}\left(\bigvee_{t\in(t_0,T]} x(t) \notin \mathcal{X}_s\right). \tag{3}$$

2.4.3 Exit Times

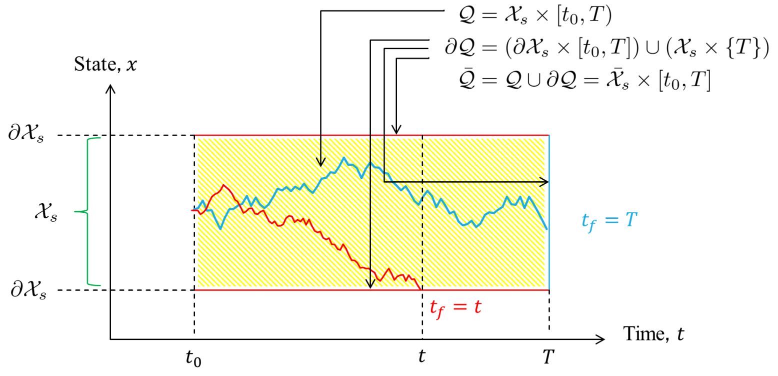
Let $\mathcal{Q} = \mathcal{X}_s \times [t_0, T)$ be a bounded set with the boundary $\partial\mathcal{Q} = (\partial\mathcal{X}_s \times [t_0, T]) \cup (\mathcal{X}_s \times \{T\})$, and closure $\overline{\mathcal{Q}} = \mathcal{Q} \cup \partial\mathcal{Q} = \overline{\mathcal{X}_s} \times [t_0, T]$. We define exit time t_f for process $x(t)$ as

$$t_f \coloneqq \inf\{t > t_0 : (x(t), t) \notin \mathcal{Q}\}.$$

Alternatively, t_f can be defined as

$$t_f \coloneqq \begin{cases} T, & \text{if } \ x(t) \in \mathcal{X}_s, \forall t \in (t_0, T), \\ \inf\left\{t \in (t_0, T) : x(t) \notin \mathcal{X}_s\right\}, & \text{otherwise.} \end{cases} \tag{4}$$

For one-dimensional state space, the domains $\mathcal{X}_s$, $\mathcal{Q}$ and their boundaries $\partial\mathcal{X}_s$, $\partial\mathcal{Q}$ are shown in Figure 2. The figure also depicts the exit times for two realizations of trajectories $\{x(t), t \in [t_0, t_f]\}$.



$\mathcal{Q} = \mathcal{X}_s \times [t_0, T)$
$\partial\mathcal{Q} = (\partial\mathcal{X}_s \times [t_0, T]) \cup (\mathcal{X}_s \times \{T\})$
$\bar{\mathcal{Q}} = \mathcal{Q} \cup \partial\mathcal{Q} = \bar{\mathcal{X}}_s \times [t_0, T]$
State, x
$\partial\mathcal{X}_s$
$\mathcal{X}_s$
$\partial\mathcal{X}_s$
$t_f = T$
$t_f = t$
Time, t
t_0
t
T

Figure 2: Computational domains and exit times t_f

Similarly, exit time $\hat{t}_f$ for process $\hat{x}(t)$ is defined as

$$\hat{t}_f \coloneqq \inf\{t > t_0 : (\hat{x}(t), t) \notin \mathcal{Q}\}. \tag{5}$$

Note that by the above definitions, $(x(t_f), t_f) \in \partial\mathcal{Q}$ and $\left(\hat{x}(\hat{t}_f), \hat{t}_f\right) \in \partial\mathcal{Q}$.

2.5 Problem Formulation

We first formalize a chance-constrained SOC problem in Section 2.5.1. In Section 2.5.2, we formulate its dual.

2.5.1 Chance-Constrained SOC Problem

Consider a cost function that is quadratic in the control input and has the form:

$$C\left(x_0, t_0, u(\cdot)\right) \coloneqq \mathbb{E}_{x_0,t_0}\left[\psi\left(x(t_f)\right) \cdot \mathbb{1}_{x(t_f)\in\mathcal{X}_s} + \int_{t_0}^{t_f}\left(\frac{1}{2}u^\top R\left(x(t), t\right)u + V\left(x(t), t\right)\right)dt\right]$$

where $\psi\left(x(t_f)\right)$ denotes a terminal cost, $V\left(x(t), t\right)$ a state dependent running cost, and $R\left(x(t), t\right) \in \mathbb{R}^{m\times m}$ a given positive definite matrix (for all values of $x(t)$ and t). $\mathbb{1}_{x(t_f)\in\mathcal{X}_s}$ returns 1 if the state of the system at the exit time t_f is inside the safe region and 0 otherwise i.e., the terminal cost is active only when the state of the system at the exit time

t_f is safe. Note that our cost function is defined over time horizon $[t_0, t_f]$ instead of $[t_0, T]$, because we do not consider the cost incurred after the system fails. We wish to find an optimal control policy for system (1) such that $C(x_0, t_0, u(\cdot))$ is minimal and the probability of failure (3) is below a specified threshold. This problem can be formulated as a chance-constrained SOC problem as follows:

Problem 1 (Chance-constrained SOC problem)**.**

$$\begin{aligned} &\min_{u(\cdot)} \mathbb{E}_{x_0,t_0}\Big[\psi\left(x(t_f)\right)\cdot \mathbb{1}_{x(t_f)\in\mathcal{X}_s} + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top Ru + V\Big)dt\Big] \\ &s.t. \;\; dx = fdt + Gudt + \Sigma dw, \quad x(t_0) = x_0, \\ &\qquad P_{x_0,t_0}\left(\bigvee_{t\in(t_0,T]} x(t)\notin\mathcal{X}_s\right) \le \Delta, \end{aligned} \tag{6}$$

where $\Delta \in (0,1)$ represents a given risk tolerance over the horizon $[t_0, T]$, and the admissible policy $u(\cdot)$ is measurable with respect to the σ-algebra generated by $x(s), 0 \le s \le t$.

2.5.2 Dual SOC Problem

We define the Lagrangian associated with Problem 1 as

$$\mathcal{L}\left(x_0, t_0, u(\cdot); \eta\right) + = C\left(x_0, t_0, u(\cdot)\right) + \eta P_{x_0,t_0}\left(\bigvee_{t\in(t_0,T]} x(t)\notin\mathcal{X}_s\right) - \eta\Delta \tag{7}$$

where $\eta \ge 0$ is the *Lagrange multiplier*. Using a standard equality in probability theory [1], P_{fail} in the chance constraint of (6) can be transformed into an expectation of an indicator function as

$$P_{x_0,t_0}\left(\bigvee_{t\in(t_0,T]} x(t)\notin\mathcal{X}_s\right) = \mathbb{E}_{x_0,t_0}\left[\mathbb{1}_{x(t_f)\in\partial\mathcal{X}_s}\right]. \tag{8}$$

Here, $\mathbb{1}_{x(t_f)\in\partial\mathcal{X}_s}$ returns 1 if the state of the system (1) at the exit time is on the boundary of the safe set $\partial\mathcal{X}_s$ (i.e. the state of the system at the exit time escapes the the safe region) and 0, otherwise. Using (8), the Lagrangian (7) can be reformulated as

$$\mathcal{L}(x_0, t_0, u(\cdot); \eta) = C(x_0, t_0, u(\cdot)) + \eta\left[\mathbb{E}_{x_0,t_0}\left[\mathbb{1}_{x(t_f)\in\partial\mathcal{X}_s}\right] - \Delta\right] \tag{9}$$

Observe that for any $\eta \ge 0$ if we define $\phi : \overline{\mathcal{X}_s} \to \mathbb{R}$ as[1]:

$$\phi\left(x;\eta\right) \coloneqq \psi\left(x\right)\cdot\mathbb{1}_{x\in\mathcal{X}_s} + \eta\cdot\mathbb{1}_{x\in\partial\mathcal{X}_s} - \eta\Delta, \tag{10}$$

then, the second term in (9) can be absorbed in a new terminal cost function ϕ as follows:

$$\mathcal{L}(x_0,t_0,u(\cdot);\eta) = \mathbb{E}_{x_0,t_0}\Big[\phi(x(t_f);\eta) + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top Ru + V\Big)dt\Big]. \tag{11}$$

Now we formulate the dual problem as follows:

Problem 2 (Dual SOC problem)**.**

$$\begin{aligned} &\max_{\eta}\min_{u(\cdot)} \;\; \mathcal{L}\left(x_0, t_0, u(\cdot); \eta\right) \\ &\qquad s.t. \;\; \eta \ge 0 \\ &\qquad\qquad dx = fdt + Gudt + \Sigma dw, \quad x(t_0) = x_0, \end{aligned} \tag{12}$$

[1]In what follows, function $\phi(x;\eta)$ sets a boundary condition for a PDE and we often need technical assumptions on the regularity of $\phi(x;\eta)$ (e.g., continuity on $\overline{\mathcal{X}_s}$) to guarantee the existence of a solution of the PDE. When such requirements are needed, we approximate (10) as $\phi(x;\eta) \approx \psi(x)B(x) + \eta\left(1 - B(x)\right)$, where $B(x)$ is a smooth bump function on $\mathcal{X}_s$.

In order to solve Problem 2 we first solve the subproblem

$$g(\eta) \coloneqq \min_{u(\cdot)} \mathcal{L}\left(x_0, t_0, u(\cdot); \eta\right). \tag{13}$$

Note that, $\mathcal{L}$ possesses the time-additive Bellman structure i.e., for any $\eta \geq 0$ and $t_0 \leq t \leq t_f$, we can write

$$\mathcal{L}\left(x_0, t_0, u(\cdot); \eta\right) = \mathbb{E}_{x_0,t_0}\left[\int_{t_0}^{t}\left(\frac{1}{2}u^\top Ru + V\right)dt\right] + \mathbb{E}_{x,t}\left[\phi\left(x(t_f); \eta\right) + \int_{t}^{t_f}\left(\frac{1}{2}u^\top Ru + V\right)dt\right].$$

Therefore problem (13) can be solved by utilizing dynamic programming without having to introduce any conservative approximation of the failure probability P_{fail}. After solving the subproblem (13) we solve the dual problem

$$\max_{\eta\geq 0} g(\eta). \tag{14}$$

Since the dual function $g(\eta)$ is the pointwise infimum of a family of affine functions of η, it is concave even when the primal problem is not convex. Moreover, since affine functions are upper semicontinuous, $g(\eta)$ is also upper semicontinuous.

2.6 Synthesis of Optimal Control Policies

In this section, we first express the dual function in terms of the solution to an HJB PDE parametrized by the dual variable η. Next, we show that the strong duality holds between the primal problem (6) and the dual problem (12) (the duality gap is zero) under a certain assumption on the system dynamics and the cost function. Finally, we propose a novel Monte-Carlo-based dual ascent algorithm to numerically solve the dual problem (14). This implies that the optimal control input for the original chance-constrained problem (primal problem) can be computed online by real-time Monte-Carlo simulations.

2.6.1 Computation of the Dual Function

In this section, we compute the dual function by solving problem (13) using dynamic programming. For each $(x,t) \in \overline{\mathcal{Q}}$, $\eta \geq 0$ and an admissible policy $u(\cdot)$ over $[t, T)$, we define the cost-to-go function:

$$\mathcal{L}(x,t,u(\cdot);\eta) = \mathbb{E}_{x,t}\left[\phi\left(x(t_f);\eta\right) + \int_t^{t_f}\left(\frac{1}{2}u^\top Ru + V\right)dt\right].$$

Now, we state the following theorem.

Theorem 1. *Suppose for a given $\eta \geq 0$, there exists a function $J : \overline{\mathcal{Q}} \to \mathbb{R}$ such that*

(a) $J(x,t;\eta)$ is continuously differentiable in t and twice continuously differentiable in x in the domain $\mathcal{Q}$;

(b) $J(x,t;\eta)$ solves the following dynamic programming PDE (HJB PDE):

$$\begin{cases} -\partial_t J = -\frac{1}{2}(\partial_x J)^\top G R^{-1} G^\top \partial_x J + V + f^\top \partial_x J + \frac{1}{2}Tr\left(\Sigma\Sigma^\top \partial_x^2 J\right), & \forall (x,t) \in \mathcal{Q}, \\ \lim\limits_{\substack{(x,t)\to(y,s) \\ (x,t)\in\mathcal{Q}}} J(x,t;\eta) = \phi(y;\eta), & \forall (y,s) \in \partial\mathcal{Q}. \end{cases} \tag{15}$$

Then, the following statements hold:

(i) $J(x,t;\eta)$ is the value function for Problem (13). That is,

$$J\left(x,t;\eta\right) = \min_{u(\cdot)} \mathcal{L}\left(x,t,u(\cdot);\eta\right), \quad \forall\, (x,t) \in \overline{\mathcal{Q}}.$$

(ii) The solution to Problem (13) is given by

$$u^*(x,t;\eta) = -R^{-1}\left(x,t\right) G^\top\left(x,t\right) \partial_x J\left(x,t;\eta\right). \tag{16}$$

■

Proof. Let J be the function satisfying (a) and (b). By Dynkin's formula [59, Theorem 7.4.1], for each $(x,t) \in \overline{\mathcal{Q}}$ and $\eta \geq 0$, we have

$$\mathbb{E}_{x,t}\left[J\left(x(t_f), t_f; \eta\right)\right] = J(x,t;\eta) + \mathbb{E}_{x,t}\left[\int_t^{t_f} dJ\left(x(s), s; \eta\right)\right], \tag{17}$$

where

$$\begin{aligned} dJ\left(x(s), s; \eta\right) =& (\partial_t J) ds + (dx)^\top \partial_x J + \frac{1}{2}(dx)^\top (\partial_x^2 J)(dx) \\ =& (\partial_t J) ds + (f + Gu)^\top (\partial_x J) ds + (\Sigma dw)^\top (\partial_x J) + \frac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right) ds. \end{aligned} \tag{18}$$

Notice that the term $\int_t^{t_f} (\Sigma dw)^\top (\partial_x J)$ is an Itô integral. Using the property of Itô integral [59, Chapter 3], we get

$$\mathbb{E}_{x,t} \int_t^{t_f} (\Sigma dw)^\top (\partial_x J) = 0 \tag{19}$$

Substituting (18) into (17) and using (19), we have

$$\mathbb{E}_{x,t}\left[J\left(x(t_f), t_f; \eta\right)\right] = J(x,t;\eta) + \mathbb{E}_{x,t}\left[\int_t^{t_f} \left(\partial_t J + (f + Gu)^\top (\partial_x J) + \frac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right)\right) ds\right]. \tag{20}$$

By the boundary condition of the PDE (15),
$J\left(x(t_f), t_f; \eta\right) = \phi\left(x(t_f); \eta\right)$. Hence, from (20), we obtain

$$J(x,t;\eta) = \mathbb{E}_{x,t}\left[\phi\left(x(t_f); \eta\right)\right] - \mathbb{E}_{x,t}\left[\int_t^{t_f} \left(\partial_t J + (f + Gu)^\top (\partial_x J) + \frac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right)\right) ds\right]. \tag{21}$$

Now, notice that the right hand side of the PDE in (15) can be expressed as the minimum value of a quadratic form in u as follows:

$$\begin{aligned} -\partial_t J &= -\frac{1}{2}(\partial_x J)^\top G R^{-1} G^\top \partial_x J + V + f^\top \partial_x J + \frac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right) \\ &= \min_u \left[\frac{1}{2}u^\top R u + V + (f + Gu)^\top \partial_x J + \frac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right)\right]. \end{aligned}$$

Therefore, for an arbitrary u, we have

$$-\partial_t J \leq \frac{1}{2}u^\top R u + V + (f + Gu)^\top \partial_x J + \frac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right) \tag{22}$$

where the equality holds iff

$$u = -R^{-1} G^\top \partial_x J. \tag{23}$$

Combining (21) and (22), we obtain

$$\begin{aligned} J(x,t;\eta) &\leq \mathbb{E}_{x,t}[\phi(x(t_f);\eta)] + \mathbb{E}_{x,t}\left[\int_t^{t_f} \left(\frac{1}{2}u^\top R u + V\right) ds\right] \\ &= \mathcal{L}\left(x, t, u(\cdot); \eta\right). \end{aligned} \tag{24}$$

This proves the statement (i). Since (24) holds with equality iff (23) is satisfied, the statement (ii) also follows. $\square$

Theorem 1 implies that the solution of problem (13) can be expressed in terms of the HJB PDE (15) parameterized by the dual variable η. Notice that (15) is a nonlinear PDE, which is in general, difficult to solve. In what follows, we linearize the PDE (15) whose solution can be obtained by utilizing the Feyman-Kac lemma [6]. First, we make the following assumption which is essential to linearize the PDE (15).

Assumption 1. *For all $(x,t) \in \overline{\mathcal{Q}}$, there exists a positive constant λ satisfying the following equation:*

$$\Sigma(x,t)\Sigma^\top(x,t) = \lambda G(x,t) R^{-1}(x,t) G^\top(x,t). \tag{25}$$

The above assumption implies that the control input in the direction with higher noise variance is cheaper than that in the direction with lower noise variance. See [2] for further discussion on this condition. Suppose Assumption 1 holds. Using the constant λ that satisfies (25), we introduce the following transformed value function $\xi(x,t;\eta)$:

$$J(x,t;\eta) = -\lambda \log\left(\xi\left(x,t;\eta\right)\right). \tag{26}$$

Transformation (26) allows us to write the PDE (15) in terms of $\xi\left(x,t;\eta\right)$ as:

$$\begin{cases} \partial_t\xi = \dfrac{V\xi}{\lambda} - \dfrac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top\partial_x^2\xi\right) + \dfrac{1}{2\xi}(\partial_x\xi)^T\Sigma\Sigma^\top\partial_x\xi - \dfrac{\lambda}{2\xi}(\partial_x\xi)^\top\left(GR^{-1}G^\top\right)\partial_x\xi - f^\top\partial_x\xi, & \forall(x,t)\in\mathcal{Q}, \\ \lim\limits_{\substack{(x,t)\to(y,s)\\(x,t)\in\mathcal{Q}}} \xi(x,t;\eta) = \exp\left(-\frac{\phi(y;\eta)}{\lambda}\right), & \forall(y,s)\in\partial\mathcal{Q}. \end{cases} \tag{27}$$

Using Assumption 1 in the equation (27), we rewrite PDE (15) as a linear PDE in terms of $\xi\left(x,t;\eta\right)$ as:

$$\begin{cases} \partial_t\xi = \frac{V\xi}{\lambda} - f^\top\partial_x\xi - \frac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top\partial_x^2\xi\right), & \forall(x,t)\in\mathcal{Q}, \\ \lim\limits_{\substack{(x,t)\to(y,s)\\(x,t)\in\mathcal{Q}}} \xi(x,t;\eta) = \exp\left(-\frac{\phi(y;\eta)}{\lambda}\right), & \forall(y,s)\in\partial\mathcal{Q}. \end{cases} \tag{28}$$

Now we find the solution of the linearized PDE (28) using the Feynman-Kac lemma.

Lemma 1 (Feynman-Kac lemma)**.** *At any $\eta \geq 0$, the solution to the linear PDE (28) exists. Moreover, the solution is unique in the sense that ξ solving (28) is given by*

$$\xi(x,t;\eta) = \mathbb{E}_{x,t}\left[exp\left(-\frac{\phi\left(\hat{x}(\hat{t}_f);\eta\right)}{\lambda} - \frac{1}{\lambda}\int_t^{\hat{t}_f} V(\hat{x}(r),r)dr\right)\right]. \tag{29}$$

where $\hat{x}(t)$ evolves according to the dynamics (2) starting at (x,t) and the exit time $\hat{t}_f$ is defined according to (5).

Proof. The proof follows from [59, Theorem 9.1.1]. □

Now we state the following theorem which proves that under Assumption 1, for a given value of η, the optimal policy of problem (13) exists and is unique.

Theorem 2. *If there exists a positive constant λ that satisfies Assumption 1, then for any $\eta \geq 0$ a unique value function $J(x,t;\eta)$ of the Problem (13) exists and is given by $J(x,t;\eta) = -\lambda\, log\left(\xi\left(x,t;\eta\right)\right)$ where $\xi(x,t;\eta)$ is given by (29). Consequently, there exists a unique optimal policy given by (16).*

Proof. According to Lemma 1, the solution of the linear PDE (28) exists and is unique in the sense that ξ solving (28) is given by (29). Therefore $J(x,t;\eta)$ is unique and is given by (26). Consequently, from Theorem 1, a unique optimal policy exists and is given by (16). □

2.6.2 Strong Duality

Due to the weak duality [60], the value of the dual problem (14) is always a lower bound for the primal problem (6). The difference between the values of (14) and (6) is called the duality gap. When the duality gap is zero, we say that the strong duality holds. In our current study on chance-constrained SOC, strong duality has a practical significance as it implies that an optimal solution to the "hard-constrained" problem (6) can be obtained by solving the "soft-constrained" problem (13), provided that the dual variable η is properly chosen.

Since (6) is a non-convex optimization problem in general, establishing the strong duality between (6) and (14) is not trivial. In Appendix 11.1, we outline general conditions under which a non-convex optimization problem admits a zero duality gap. In the sequel, we apply the result in Appendix 11.1 to (6) to delineate the conditions under which the chance-constrained SOC admits the strong duality.

The following assumption is reminiscent of the Slater's condition, which is often a natural premise for strong duality:

Assumption 2. *There exists a policy $\widetilde{u}(\cdot)$ such that $P_{fail}(x_0,t_0,\widetilde{u}(\cdot)) - \Delta < 0$, i.e., Problem 1 is strictly feasible.*

Using Lemma 3 in the Appendix 11.1, we can show that under Assumptions 1 and 2, there exists a dual optimal solution η^* such that $0 \leq \eta^* < \infty$ such that $g(\eta^*) = \max_{\eta \geq 0} g(\eta)$. To proceed further, we need the following assumption. We conjecture that this assumption is valid under mild conditions; a formal analysis is postponed as future work.

Assumption 3. *Suppose Assumption 1 holds so that for each $\eta \geq 0$ a unique optimal policy $u^*(x,t;\eta)$ of Problem (13) exists (c.f., Theorem 2). We further assume that the function $\eta \mapsto P_{\text{fail}}(x_0, t_0, u^*(x,t;\eta)) : [0,\infty) \to [0,1]$ is continuous.*

Now notice that under Assumption 1, the solution $u^*(\cdot;\eta)$ of the problem $\operatorname{argmin}_{u(\cdot)} \mathcal{L}(x_0, t_0, u(\cdot);\eta)$ exists and is unique for each $\eta \geq 0$. Therefore, statements (i) and (ii) of the Assumption 8 in the Appendix 11.1 hold true. Moreover, by the Assumption 3, statement (iii) of Assumption 8 in the Appendix 11.1 is satisfied. Now under Assumption 8 using Lemma 4, we can prove the following complementary slackness statements:

(a) If $\eta^* = 0$, then $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,;\eta^*) - \Delta \leq 0$

(b) If $\eta^* > 0$, then $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,;\eta^*)) - \Delta = 0$

The following theorem is the main result of this section.

Theorem 3. *Consider problem 1 and suppose Assumptions 1, 2 and 3 hold. Then there exists a dual optimal solution $0 \leq \eta^* < \infty$ that maximizes $g(\eta)$ and a unique optimal policy $u^*(\cdot\,;\eta^*)$ of problem (13) is an optimal policy of Problem 1 (primal problem) such that $C(x_0, t_0, u^*(\cdot\,;\eta^*)) = g(\eta^*)$ i.e., the duality gap is zero.*

Proof. Refer to the proof of Theorem 17 in the Appendix 11.1. □

Similar to our approach, the work presented in [44] solves the chance-constrained problem by formulating its dual. However, in this work, the joint chance constraint is conservatively approximated using Boole's inequality. Hence, the duality gap is nonzero and the obtained solution is suboptimal. Unlike this method, in our work, we prove that the strong duality exists between the primal chance-constrained problem (6) and its dual (12) under certain assumptions on the system dynamics and cost function. Consequently, the chance-constrained problem can be solved by evaluating the dual objective function (c.f. Section 2.6.1) and solving the dual problem (c.f. Section 2.6.5).

In order to solve the dual problem, it is natural to use the gradient ascent algorithm $\eta \leftarrow \eta + \gamma(P_{\text{fail}}(x_0, t_0, u^*(\cdot;\eta)))$ to iteratively update the dual variable η. Here, γ is the step size, $u^*(.;\eta)$ is the optimal policy solving (13), and $P_{\text{fail}}(x_0, t_0, u^*(\cdot;\eta))$ is the probability of failure under the policy $u^*(.;\eta)$. For the dual ascent algorithm to be practical, we need to be able to evaluate the gradient $P_{\text{fail}} - \Delta$ efficiently in each iteration. Therefore, next, we study how to compute P_{fail} for the given value of η.

2.6.3 Risk Estimation

We present two approaches to compute $P_{\text{fail}}(x_0, t_0, u^*(.;\eta))$ for a given value of η. The first approach is PDE-based in which we find the optimal policy $u^*(.;\eta)$ first, and then evaluate its risk $P_{\text{fail}}(x_0, t_0, u^*(.;\eta))$ by solving a PDE. Despite conceptual simplicity, this approach is difficult to implement computationally unless there exists an analytical expression of $u^*(.;\eta)$. To circumvent this difficulty, we also present an importance-sampling-based approach. This approach allows us to numerically compute $P_{\text{fail}}(x_0, t_0, u^*(.;\eta))$ without ever constructing $u^*(.;\eta)$ and is more computationally amenable.

PDE-Based Approach:

We state the following theorem:

Theorem 4. *Suppose there exists a function $J : \overline{\mathcal{Q}} \to \mathbb{R}$ such that*

(a) $J(x,t)$ is continuously differentiable in t and twice continuously differentiable in x in the domain $\mathcal{Q}$;

(b) $J(x,t)$ solves the following PDE for the given optimal control policy $u^*(\cdot)$:

$$\begin{cases} -\partial_t J = (f + Gu^*)^\top \partial_x J + \frac{1}{2} Tr\big(\Sigma\Sigma^\top \partial_x^2 J\big), & \forall (x,t) \in \mathcal{Q}, \\ \lim_{\substack{(x,t)\to(y,s) \\ (x,t)\in\mathcal{Q}}} J(x,t) = \phi'(y), & \forall (y,s) \in \partial\mathcal{Q}. \end{cases} \tag{30}$$

where $\phi'(x) = \mathbb{1}_{x\in\partial\mathcal{X}_s}$. Then, P_{fail} is given by

$$P_{\text{fail}}\left(x_0, t_0, u^*(\cdot)\right) = J(x_0, t_0).$$

Proof. Let $J(x,t)$ be the function satisfying (a) and (b). By Dynkin's formula, for each $(x,t) \in \overline{\mathcal{Q}}$ we have

$$\mathbb{E}_{x,t}\left[J\left(x(t_f), t_f\right)\right] = J(x,t) + \mathbb{E}_{x,t}\left[\int_t^{t_f}\left(\partial_t J + (f + Gu^*)^\top(\partial_x J) + \frac{1}{2}\text{Tr}\left(\Sigma\Sigma^\top\partial_x^2 J\right)\right)ds\right].$$

The second term on the right side contains, in parentheses, the PDE in (30) and is therefore zero. Hence, we obtain

$$J(x,t) = \mathbb{E}_{x,t}\left[J\left(x(t_f), t_f\right)\right]. \tag{31}$$

From the boundary condition of the PDE (30)

$$\mathbb{E}_{x,t}[J(x(t_f), t_f)] = \mathbb{E}_{x,t}\left[\phi'\left(x(t_f)\right)\right] = \mathbb{E}_{x,t}\left[\mathbb{1}_{x(t_f)\in\partial\mathcal{X}_s}\right]. \tag{32}$$

Combining (31), (32), and we obtain

$$J(x_0, t_0) = P_{\text{fail}}\left(x_0, t_0, u^*(\cdot)\right).$$

□

Remark 1. *We do not prove here the existence of the solution of the PDE (30), rather we suppose that a solution exists. Note that PDE (30) is a special case of the Cauchy-Dirichlet problem. For proof of the existence of a solution to the Cauchy-Dirichlet problem, we refer the readers to [61, Chapter 6].*

Theorem 4 implies that if we have the solution for the optimal policy $u^*(\cdot)$, $P_{\text{fail}}\left(x_0, t_0, u^*(\cdot)\right)$ can be computed by solving the PDE (30). Next, we present an importance-sampling-based approach to find P_{fail} without constructing $u^*(\cdot)$.

Importance-Sampling-Based Approach:

Let $\mathcal{T}$ be the space of trajectories $x := \{x(t), t \in [t_0, t_f]\}$. Let $Q^*(x)$ be the probability distribution of the trajectories defined by system (1) under the optimal policy $u^*(\cdot)$. Using (8), we can write

$$P_{\text{fail}}(x_0, t_0, u^*(\cdot)) = \int_{\mathcal{T}} \mathbb{1}_{x(t_f)\in\partial\mathcal{X}_s} Q^*(dx). \tag{33}$$

Suppose we generate an ensemble of large number of N trajectories $\{x^{(i)}\}_{i=1}^N$ under the distribution Q^*. Then according to the strong law of large numbers, as $N \to \infty$,

$$\frac{1}{N}\sum_{i=1}^N \mathbb{1}_{x^{(i)}(t_f)\in\partial\mathcal{X}_s} \xrightarrow{a.s.} P_{\text{fail}}(x_0, t_0, u^*(\cdot)) \quad x^{(i)} \sim Q^*(x).$$

This implies that a Monte Carlo algorithm is applicable to numerically evaluate P_{fail}. However, such a Monte Carlo algorithm is impractical since it is difficult to sample trajectories from Q^* as the optimal policy u^* is unknown. Fortunately, an importance sampling scheme is available which allows us to numerically evaluate P_{fail} using a trajectory ensemble sampled from the distribution $P(x)$ of the uncontrolled system (2). The next theorem provides the details:

Theorem 5. *Suppose we generate an ensemble of a large number of N trajectories $\{x^{(i)}\}_{i=1}^N$ under the distribution P. For each i, let $r^{(i)}$ be the path reward of the sample path i given by*

$$r^{(i)} = exp\left(-\frac{\phi\left(x^{(i)}(t_f); \eta\right)}{\lambda} - \frac{1}{\lambda}\int_t^{t_f} V\left(x^{(i)}(r), r\right)dr\right) \tag{34}$$

Let us define $r := \sum_{i=1}^N r^{(i)}$. Then as $N \to \infty$,

$$\sum_{i=1}^N \frac{r^{(i)}}{r}\mathbb{1}_{x^{(i)}(t_f)\in\partial\mathcal{X}_s} \xrightarrow{a.s.} P_{\text{fail}}(x_0, t_0, u^*(\cdot)) \quad x^{(i)} \sim P(x).$$

Proof. Notice that P_{fail} in (33) can be equivalently written as

$$P_{\text{fail}}(x_0, t_0, u^*(\cdot)) = \int_{\mathcal{T}} \mathbb{1}_{x(t_f)\in\partial\mathcal{X}_s} \frac{dQ^*}{dP}(x) P(dx) \tag{35}$$

where the Radon-Nikodym derivative $\frac{dQ^*}{dP}(x)$ represents the likelihood ratio of observing a sample path x under distributions Q^* and P. Now, according to Theorem 19 in the Appendix 11.3, for a given ensemble of N trajectories $\{x^{(i)}\}_{i=1}^N$ sampled under the distribution P, the likelihood ratio $\frac{dQ^*}{dP}$ of observing a sample path $x^{(i)}$ is given by $\frac{r^{(i)}/r}{1/N}$. Therefore, using (35), by the strong law of large numbers, as $N \to \infty$ we get

$$\begin{aligned}&\frac{1}{N}\sum_{i=1}^{N} \frac{r^{(i)}/r}{1/N} \mathbb{1}_{x^{(i)}(t_f)\in\partial\mathcal{X}_s} \overset{a.s.}{\to} P_{\text{fail}}(x_0, t_0, u^*(\cdot))\\ &x^{(i)} \sim P(x)\end{aligned}$$

which completes the proof. $\square$

Theorem 5 provides us a sampling-based approach to numerically compute the probability of failure $P_{\text{fail}}(x_0, t_0, u^*(\cdot))$. According to Theorem 5, if we sample N trajectories under distribution P, then we can approximate the probability of failure as

$$P_{\text{fail}}(x_0, t_0, u^*(\cdot)) \approx \sum_{i=1}^{N} \frac{r^{(i)}}{r} \mathbb{1}_{x^{(i)}(t_f)\in\partial\mathcal{X}_s} \quad x^{(i)} \sim P(x). \tag{36}$$

where $r^{(i)}$ is defined by (34) and $r = \sum_{i=1}^N r^{(i)}$. Note that sampling under distribution P is easy since we only need to simulate the uncontrolled system dynamics (2).

2.6.4 Generalized Risk-Estimation PDE

In select applications, requiring that the solution to (30) satisfies the boundary condition everywhere on $\partial\mathcal{Q}$ can be overly restrictive. In this section, we consider a generalized version of PDE (30) by reducing the requirement in the boundary condition to hold only for a subset of points on the boundary $(x, t) \in \partial\mathcal{Q}$ called the *regular points*. We show that if a solution of such a PDE exists, then we can still interpret it as the probability of failure. Before we state the generalized version of Theorem 4 we need the following definitions and an auxiliary lemma.

Definition 2 (Regular and irregular points)**.** *A point $(x, t) \in \partial\mathcal{Q}$ is called regular for $\mathcal{Q}$ with respect to $x(t)$ if $P_{x,t}(t_f = t_0) = 1$; i.e., a.a. paths $x(t)$ starting from x at time t leave $\mathcal{Q}$ immediately. Otherwise the point (x, t) is called irregular.*

Example 1. *Consider a punctured ball $\mathcal{B} = \{x \in \mathbb{R}^d : 0 < \|x\| < 1\}$ and its boundary $\partial\mathcal{B} = \{x \in \mathbb{R}^d : \|x\| = 1\} \cup \{\mathbf{0}\}$ i.e, the boundary $\partial\mathcal{B}$ contains the surface of the ball $\mathcal{B}$ and the origin. If a Brownian motion $w(t)$ starts at the origin, it returns to $\mathcal{B}$ immediately. On the other hand, if it starts anywhere on the surface of $\mathcal{B}$, it leaves $\mathcal{B}$ immediately (c.f., the 0-1 law [59, Corollary 9.2.7]). Hence, the origin is irregular for $\mathcal{B}$ with respect to $w(t)$, whereas all points on the surface of $\mathcal{B}$ are regular.*

For more interesting examples of regular and irregular points, see [62, Chapter 4].

Definition 3 (Hunt's condition)**.** *A process $x(t)$ is said to satisfy Hunt's condition if every semipolar set for process $x(t)$ is also polar for $x(t)$. A semipolar set is a countable union of thin sets and a measurable set E is called thin for process $x(t)$ if for all starting points, $x(t)$ does not hit E immediately, a.s. A measurable set F is called polar for process $x(t)$ if for all starting points, $x(t)$ never hits F, a.s.*

Hunt's condition holds for Brownian motion [63]. See [59] for a discussion on the requirements on the coefficients of the SDE (1) in order for it to satisfies Hunt's condition.

Lemma 2. *Let $\mathcal{Q}^I$ denote the set of irregular points of $\mathcal{Q}$ with respect to process $x(t)$. Suppose $x(t)$ satisfies the Hunt's condition. Then $(x(t_f), t_f) \notin \mathcal{Q}^I$ a.s.*

Proof. From the definition of irregular points, the set $\mathcal{Q}^I$ is semipolar for $x(t)$. Furthermore, since $x(t)$ satisfies Hunt's condition, the set $\mathcal{Q}^I$ is polar for $x(t)$, and therefore $(x(t_f), t_f) \notin \mathcal{Q}^I$ a.s. $\square$

We can now state the generalized version of Theorem 4 as follows:

Theorem 6. *Suppose $x(t)$ satisfies Hunt's condition and there exists a function $J : \overline{\mathcal{Q}} \to \mathbb{R}$ such that*

(a) $J(x,t)$ is continuously differentiable in t and twice continuously differentiable in x in domain $\mathcal{Q}$;

(b) $J(x,t)$ solves the following PDE for a given admissible control policy $u(\cdot)$:

$$\begin{cases} -\partial_t J = (f+Gu)^T \partial_x J + \frac{1}{2} Tr\big(\Sigma\Sigma^T \partial_x^2 J\big), & \forall (x,t) \in \mathcal{Q}, \\ \lim\limits_{\substack{(x,t)\to(y,s) \\ (x,t)\in\mathcal{Q}}} J(x,t) = \phi'(y), & \forall \text{ regular } (y,s) \in \partial\mathcal{Q}. \end{cases} \tag{37}$$

Then, P_{fail} is given by

$$P_{\text{fail}} = J(x_0, t_0).$$

Proof. Let $J(x,t)$ be the function satisfying (a) and (b). By Dynkin's formula, for each $(x,t) \in \overline{\mathcal{Q}}$ we have

$$\mathbb{E}_{x,t}\left[J\left(x(t_f), t_f\right)\right] = J(x,t) + \mathbb{E}_{x,t}\left[\int_t^{t_f} \left(\partial_t J + (f+Gu)^T(\partial_x J) + \frac{1}{2}\mathrm{Tr}\big(\Sigma\Sigma^T \partial_x^2 J\big)\right) ds\right].$$

The second term on the right side contains, in parentheses, the PDE in (37) and is therefore zero. Hence, we obtain

$$J(x,t) = \mathbb{E}_{x,t}\left[J\left(x(t_f), t_f\right)\right]. \tag{38}$$

Since $x(t)$ satisfies Hunt's condition, from Lemma 2 we know that $(x(t_f), t_f)$ does not belong to the irregular set of $\mathcal{Q}$ a.s. Hence from the boundary condition of PDE (37),

$$\mathbb{E}_{x,t}[J(x(t_f), t_f)] = \mathbb{E}_{x,t}\left[\phi'\left(x(t_f)\right)\right]. \tag{39}$$

Combining (38) and (39) we obtain

$$J(x_0, t_0) = P_{\text{fail}}.$$

□

Note that Theorem 6 is more useful than Theorem 4 for the risk estimation in Example 1 because the boundary condition at the origin is relaxed.

Remark 2. *PDE (37) can be considered as a special case of the generalized Dirichlet-Poisson problem, the existence of whose solution is proved in [59, Chapter 9].*

2.6.5 Solution of the Dual Problem (14)

We present two numerical approaches to solve the dual problem (14). The first approach is the finite difference method (FDM) which is one of the most popular approaches to numerically solve the partial differential equations. Despite its popularity, this approach suffers from the *curse of dimensionality* and the computational cost grows exponentially as the state-space dimension increases. To circumvent this difficulty, we also present the path integral approach. This approach is more computationally amenable. It allows us to numerically solve the dual problem (14) online via open-loop samples of system trajectories.

Finite Difference Method:

When the geometry of the computational domain is simple, it is straightforward to form discrete approximations to spatial differential operators with a high order of accuracy via Taylor series expansions [64]. A finite number of grid points are placed at the interior and on the boundary of the computational domain, and the solution to the PDE is sought at these finite set of locations. Once the grid is determined, finite difference operators are derived to approximate spatial derivatives in the PDE. In this work, centered formulas are used to approximate spatial operators with up to eight-order accuracy at the interior grid points. For grid points near the boundary, asymmetric formulas that maintain the order of accuracy are used in conjunction with Dirichlet boundary condition. The use of finite difference operators yields a system of ordinary differential equations (ODEs) that is integrated in time with the desired method. The MATLAB suite of ODE solvers [65] provides a number of efficient ODE integrators with high-temporal order, error control, and variable time-stepping that advance the solution in time.

Note that FDM numerically solves the HJB PDE (28) backward in time. The solution needs to be computed offline and the optimal control policy needs to be stored in a look-up table. For a given state and time, the optimal input for real-time control is obtained from the stored look-up table. Similar to the PDE (28), FDM can be utilized to compute the probability of failure by numerically solving the PDE (30) backward in time. It is well known that the computational complexity and memory requirements of FDM increase exponentially as the state-space dimension increases. Therefore, FDM is intractable for systems with more than a few state variables. Moreover, this method is inconvenient for real-time implementation since the HJB PDE needs to be solved backward in time. Also, FDM computes a global solution over the entire domain $\mathcal{Q}$ even if the majority of the state-time pairs (x,t) will never be visited by the actual system. To overcome these difficulties, we present the path integral approach which numerically solves the dual problem (14) in real-time.

Path Integral Approach:

Assumption 1 implies that the stochastic noise has to enter the system dynamics via the control channels. Therefore, in what follows, we assume that system (1) can be partitioned into subsystems that are directly and non-directly driven by the noise as:

$$\begin{bmatrix} dx^{(1)} \\ dx^{(2)} \end{bmatrix} = \begin{bmatrix} f^{(1)}(x,t) \\ f^{(2)}(x,t) \end{bmatrix} dt + \begin{bmatrix} \mathbf{0} \\ G^{(2)}(x,t) \end{bmatrix} u(x,t)dt + \begin{bmatrix} \mathbf{0} \\ \Sigma^{(2)}(x,t) \end{bmatrix} dw$$

where $\mathbf{0}$ denotes a zero matrix of appropriate dimensions. The path integral control framework utilizes the fact that the solution to PDE (28) admits the Feynman-Kac representation (29). The optimal control input $u^*(x,t;\eta)$ (16) can be obtained by taking the gradient of (29) with respect to x [6, 3]. We obtain

$$u^*(x,t;\eta)dt = \mathcal{G}\left(x,t\right) \frac{\mathbb{E}_{x,t}\left[\exp\left(-\frac{1}{\lambda}S\right)\Sigma^{(2)}\left(x,t\right)dw(t)\right]}{\mathbb{E}_{x,t}\left[\exp\left(-\frac{1}{\lambda}S\right)\right]}, \tag{40}$$

where matrix $\mathcal{G}\left(x,t\right)$ is defined as

$$\mathcal{G}(x,t) = R^{-1}(x,t)G^{(2)^\top}(x,t)\left(G^{(2)}(x,t)R^{-1}(x,t)G^{(2)^\top}(x,t)\right)^{-1}.$$

and S represents the cost-to-go of a trajectory of system $\hat{x}(t)$ starting at (x,t) , i.e.,

$$S = \phi\left(\hat{x}(\hat{t}_f);\eta\right) + \int_t^{\hat{t}_f} V\left(\hat{x}(t),t\right)dt.$$

To evaluate expectations in (29) and (40) numerically, we can discretize the uncontrolled dynamics (2) and use Monte Carlo sampling [6]. Unlike FDM, the path integral framework solves PDE (28) in the forward direction. It evaluates a solution locally without requiring knowledge of the solution nearby so that there is no need for a (global) discretization of the computational domain. For real-time control, Monte Carlo simulations can be performed in real-time in order to evaluate (40) for the current (x,t). Similar to (29) and (40), the failure probability (36) can be numerically computed by Monte Carlo sampling using the importance sampling method. The Monte Carlo simulations can be parallelized by using the Graphic Processing Units (GPUs) and thus the path integral approach is less susceptible to the curse of dimensionality.

Now, we present the path-integral-based dual ascent algorithm to numerically solve the dual problem (14). The algorithm for that is given in Algorithm 1. The algorithm starts by dealing with a special case $\eta = 0$. We find the optimal policy (40). Then, we compute $P_{\text{fail}}\left(x_0,t_0,u^*(\cdot;\eta=0)\right)$ by importance-sampling approach. If $P_{\text{fail}}\left(x_0,t_0,u^*(\cdot;\eta=0)\right) \leq \Delta$, we return $u^*(\cdot;\eta=0)$. Otherwise, the algorithm chooses some initial η and iteratively updates the dual variable $\eta \leftarrow \eta + \gamma(P_{\text{fail}}(x_0,t_0,u^*(\cdot;\eta)) - \Delta)$ with a learning rate of γ. Once $|P_{\text{fail}}\left(x_0,t_0,u^*(\cdot;\eta)\right) - \Delta|$ is less than the error tolerance ϵ, we return the policy $u^*(\cdot;\eta)$. Thus, using the Algorithm 1, we can numerically solve the original chance-constrained problem (6) online via real-time Monte-Carlo simulations.

2.7 Simulation Results

In this section, we demonstrate the effectiveness of the proposed control synthesis framework in addressing the chance-constrained problem defined in (6). In Section 2.7.1, we employ a 2D state-space velocity input model to solve a mobile robot navigation problem using Algorithm 1. The solution derived from the path-integral method is compared with that

Algorithm 1 Dual ascent via path integral approach

```
Require: Error tolerance $\epsilon > 0$, learning rate $\gamma > 0$
 1: Set $\eta = 0$
 2: Find $u^*(\cdot;\eta = 0)$ using (40)
 3: Compute the failure probability $P_{\text{fail}}(x_0, t_0, u^*(\cdot;\eta = 0))$ using (36).
 4: if $P_{\text{fail}}(x_0, t_0, u^*(\cdot;\eta = 0)) \leq \Delta$ then
 5:     return $u^*(\cdot;\eta = 0)$
 6: end if
 7: Choose initial $\eta > 0$.
 8: while True do
 9:     Find $u^*(\cdot;\eta)$ using (40)
10:     Compute the failure probability $P_{\text{fail}}(x_0, t_0, u^*(\cdot;\eta))$ using (36)
11:     if $|P_{\text{fail}}(x_0, t_0, u^*(\cdot;\eta)) - \Delta| < \epsilon$ then
12:         return $u^*(\cdot;\eta)$
13:     end if
14:     $\eta \leftarrow \eta + \gamma(P_{\text{fail}}(x_0, t_0, u^*(\cdot;\eta)) - \Delta)$
15: end while
```

of FDM. In Section 2.7.2, we extend the analysis to a unicycle model in a 4D state-space, and in Section 2.7.3, to a car model with a 5D state-space. In both the 4D and 5D cases, we solve the corresponding chance-constrained problems using Algorithm 1. Due to the high computational cost of applying FDM to models with 4D and 5D state spaces, we rely exclusively on the path-integral method for these examples.

2.7.1 Input Velocity Model

The problem is illustrated in Figure 3 where a particle robot wants to navigate in a 2D space from a given start position (shown by a yellow star) to the origin (shown by a magenta star), by avoiding the collisions with the red obstacle and the outer boundary.

Let the states of the system be p_x and p_y, the positions along x and y, respectively. The system dynamics are given by the following SDEs:

$$dp_x = v_x dt + \sigma dw_x, \quad dp_y = v_y dt + \sigma dw_y, \tag{41}$$

where v_x and v_y are velocities along x and y directions, respectively. Assume

$$v_x = \overline{v}_x + \widetilde{v}_x, \qquad v_y = \overline{v}_y + \widetilde{v}_y,$$

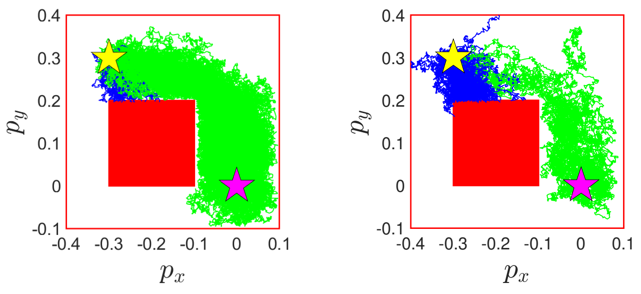
where $\overline{v}_x$ and $\overline{v}_y$ are nominal velocities given by $\overline{v}_x = -k_x p_x$ and $\overline{v}_y = -k_y p_y$ for some constants k_x and k_y. Hence, (41) can be rewritten as

$$dp_x = -k_x p_x dt + \widetilde{v}_x dt + \sigma dw_x, \quad dp_y = -k_y p_y dt + \widetilde{v}_y dt + \sigma dw_y. \tag{42}$$

Now, the goal is to design an optimal control policy $u^* = [\widetilde{v}_x^* \ \widetilde{v}_y^*]^\top$ for the chance-constrained problem (6). The initial state is $x_0 = [-0.3 \ 0.3]^\top$. We set $k_x = k_y = 0.5$, $V(x(t)) = p_x^2(t) + p_y^2(t)$, $\psi(x(T)) = p_x^2(T) + p_y^2(T)$, $R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $t_0 = 0$, $T = 2$, and $\sigma^2 = 0.01$. In Algorithm 1, we set the error tolerance $\epsilon = 0.01$ and learning rate $\gamma = 0.1$. For FDM, the computational domain is discretized by a grid of 96×96 points and the solver ode45 is used with a relative error tolerance equal to 10^{-3}. In the path integral simulation, for Monte Carlo sampling, 10^5 trajectories and a step size equal to 0.01 are used.

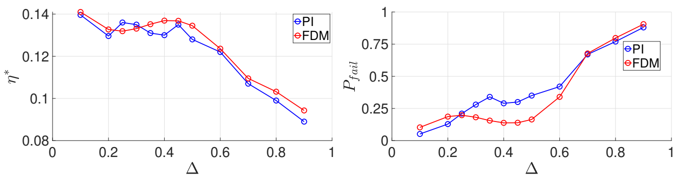
In Figure 3, we plot 100 sample trajectories generated using synthesized optimal policies for two values of Δ. The trajectories are color-coded; the blue paths collide with the obstacle or the outer boundary, while the green paths converge in the neighborhood of the origin (the goal position). For a lower value of Δ the weight of blue paths is less and that of the green paths is more as compared to the higher value of Δ. In other words, the failure probability for the lower value of Δ is less as compared to that of the higher value of Δ.

Figure 4 represents how the value of η^* and $P_{\text{fail}}(x_0, t_0, u^*(\cdot\ ;\eta^*))$ change with respect to Δ. The values obtained using path integral control and FDM are compared. We expect that the value of η^* reduces and that of P_{fail} increases as



(a) $\Delta = 0.25$ (b) $\Delta = 0.9$

Figure 3: Robot navigation problem for the input velocity model. The start position is shown by a yellow star and the target position (the origin) by a magenta star. 100 sample trajectories generated using optimal control policies for two values of Δ are shown. The trajectories are color-coded; blue paths collide with the obstacle or the outer boundary, while the green paths converge in the neighborhood of the magenta star.



(a) η^* vs Δ (b) $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,; \eta^*))$ vs Δ

Figure 4: η^* and $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,; \eta^*))$ vs Δ for input velocity model using path integral control and FDM.

Δ increases. Note that the graphs are not monotonic. This is because we used numerical methods to solve the PDE (28), which caused some errors in the computation of η^* and $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,; \eta^*))$.

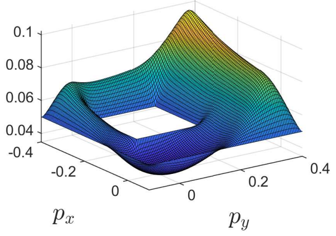
Figure 5 shows the comparison of solutions $J(x, t_0)$ and $u^*(x, t_0)$ of the PDE (15) obtained from FDM (top row) and path integral (bottom row) for $\eta = 0.05$ and $\eta = 0.13$. Since the path integral is a sampling-based approach its solutions are noisier compared to FDM, as expected. Notice that for $\eta = 0.05$, due to a lower boundary value, the control inputs $u^*(x, t_0)$ push the robot towards the obstacle or the outer boundary from the most part of the safe region $\mathcal{X}_s$, except near the origin (the target position). Whereas, for $\eta = 0.13$, aggressive inputs $u^*(x, t_0)$ are applied near the boundary to force the robot to go towards the goal position.

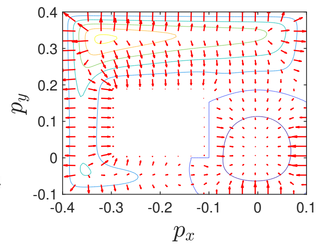
Figure 6 shows the colormaps of the failure probabilities of the synthesized optimal policies for $\Delta = 0.25$ and $\Delta = 0.9$ as functions of initial position. Notice that the region of higher failure probabilities is more for $\Delta = 0.9$ than that for the lower value of $\Delta = 0.25$.

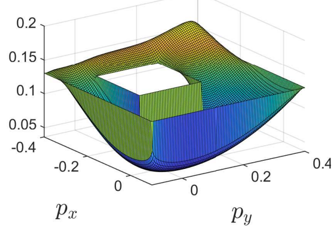
The factors that affect the computation speed of FDM include the grid size, the choice of an ODE solver and its error tolerances. Whereas the computation speed of path integral depends on the number of Monte Carlo samples and the number of time steps. The computation time is evaluated on a machine with an Intel Core i7-9750H CPU clocked at 2.6 GHz. Both FDM and path integral approaches are implemented in MATLAB. Path integral simulations are run parallelly using MATLAB's parallel processing toolbox and the GPU Nvidia GeForce GTX 1650. The average number of iterations required by Algorithm 1 to converge for both FDM and path integral is 3. We choose the initial η to be 0.05 and the learning rate γ to be 0.1. For FDM, the average computation time for each iteration is approximately 20 sec. For the path integral framework, the average computation time required for each iteration is 3 sec. Note that no special effort was made to optimize the algorithm for speed. We plan to reduce the computation time of the algorithm in future work.

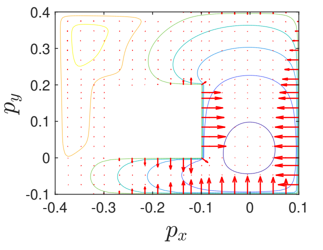
2.7.2 Unicycle Model

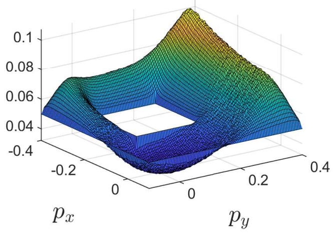
The problem is illustrated in Figure 7. Similar to the problem in Section 2.7.1, a particle robot wants to navigate in a 2D space from a given start position (shown by a yellow star) to the origin (shown by a magenta star), by avoiding the collisions with the red obstacles and the outer boundary. The states of the unicycle model $x = [p_x \; p_y \; s \;\; \theta]^\top$ consist of its

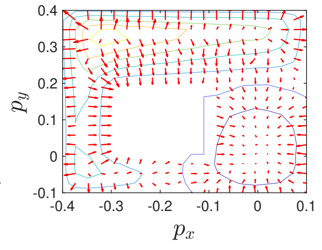

(a) $J(x,t_0)$ for $\eta = 0.05$

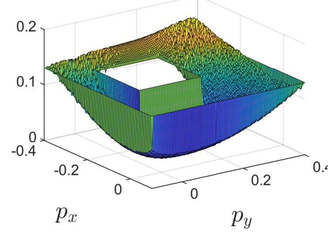

(b) $u^*(x,t_0)$ for $\eta = 0.05$


(c) $J(x,t_0)$ for $\eta = 0.13$


(d) $u^*(x,t_0)$ for $\eta = 0.13$


(e) $J(x,t_0)$ for $\eta = 0.05$


(f) $u^*(x,t_0)$ for $\eta = 0.05$


(g) $J(x,t_0)$ for $\eta = 0.13$

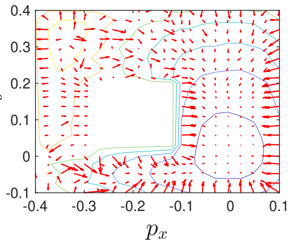
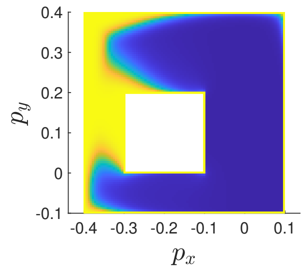

(h) $u^*(x,t_0)$ for $\eta = 0.13$

Figure 5: Input velocity model: comparison of solutions $J(x,t_0)$ and $u^*(x,t_0)$ obtained from FDM (a-d) and path integral (e-h) for $\eta = 0.05$ and $\eta = 0.13$. The optimal control inputs $u^*(x,t_0)$ in (b, d, f, h) are plotted together with contours of $J(x,t_0)$.


(a) $\Delta = 0.25$

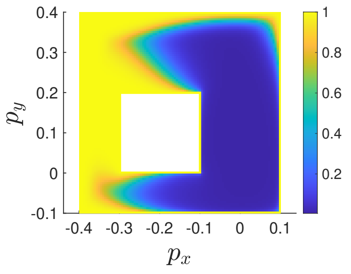

(b) $\Delta = 0.9$

Figure 6: Colormaps of the failure probabilities of the optimal policies synthesized for the input velocity model as functions of initial position.

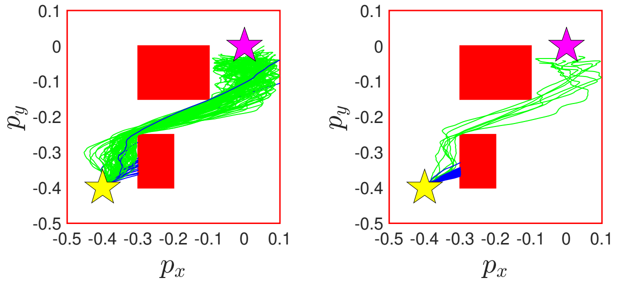
$x - y$ position $[p_x \; p_y]^\top$, speed s and heading angle $\theta \in [0, 2\pi]$. The system dynamics are given by the following SDE:

$$\begin{bmatrix} dp_x \\ dp_y \\ ds \\ d\theta \end{bmatrix} = -k \begin{bmatrix} p_x \\ p_y \\ s \\ \theta \end{bmatrix} dt + \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ 0 \\ 0 \end{bmatrix} dt + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \left(\begin{bmatrix} a \\ \omega \end{bmatrix} dt + \begin{bmatrix} \sigma & 0 \\ 0 & \nu \end{bmatrix} dw \right). \tag{43}$$

The control input $u := \begin{bmatrix} a & \omega \end{bmatrix}^\top$ consists of acceleration a and angular speed ω. σ and ν are the noise level parameters. In the simulation, we set $\sigma = \nu = 0.1$, $k = 0.2$, $t_0 = 0$, $T = 10$, $x_0 = \begin{bmatrix} -0.4 & -0.4 & 0 & 0 \end{bmatrix}^\top$, $V(x) = p_x^2 + p_y^2$ and $\psi(x(T)) = p_x^2(T) + p_y^2(T)$. We solve the chance-constrained problem (6) via path integral approach using Algorithm 1. Since the state-space of the system is of high order we do not use FDM to solve this problem. For Monte Carlo sampling, 10^4 trajectories and a step size equal to 0.01 are used.

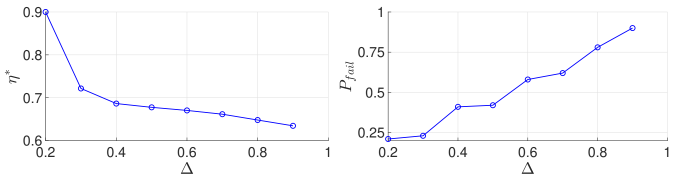
In Figure 7, we plot 100 sample trajectories generated using synthesized optimal policies for two values of Δ. The trajectories are color-coded similar to the problem in Section 2.7.1. For a lower value of Δ the weight of blue paths is less and that of the green paths is more as compared to the higher value of Δ. In other words, the failure probability for the lower value of Δ is less as compared to that of the higher value of Δ.

Figure 8 represents how the value of η^* and $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,; \eta^*))$ change with respect to Δ. As expected the value of η^* reduces and that of P_{fail} increases as Δ increases.



(a) $\Delta = 0.2$ (b) $\Delta = 0.9$

Figure 7: Robot navigation problem for the unicycle model. The start position is shown by a yellow star and the target position (the origin) by a magenta star. 100 sample trajectories generated using optimal control policies for two values of Δ are shown. The trajectories are color-coded; blue paths collide with the obstacle or the outer boundary, while the green paths converge in the neighborhood of the magenta star.



(a) η^* vs Δ (b) $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,; \eta^*))$ vs Δ

Figure 8: η^* and $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,; \eta^*))$ vs Δ for unicycle model using path integral control.

The algorithm is implemented on the same machine mentioned in Section 2.7.1. It is run in MATLAB, and the Monte Carlo simulations are run parallelly using MATLAB's parallel processing toolbox. The average number of iterations required by the proposed algorithm to converge is 4. We choose the initial η to be 0.6 and the learning rate γ to be 0.08. The average computation time required per iteration is 2.9 sec. Note that no special effort was made to optimize the algorithm for speed. We plan to reduce the computation time of the algorithm in future work.

2.7.3 Car Model

The problem is illustrated in Figure 9. A car wants to navigate in a 2D space from a given start position (shown by a yellow star) to the origin (shown by a magenta star), by avoiding the collisions with the red obstacles and the outer boundary. The states of the car model $\begin{bmatrix} p_x & p_y & s & \theta & \phi \end{bmatrix}^\top$ consists of its $x-y$ position $\begin{bmatrix} p_x & p_y \end{bmatrix}^\top$, the speed s, heading angle θ and the front wheel angle ϕ. L is the inter-axle distance. The system dynamics are given by the following SDE:

$$\begin{bmatrix} dp_x \\ dp_y \\ ds \\ d\theta \\ d\phi \end{bmatrix} = -k \begin{bmatrix} p_x \\ p_y \\ s \\ 0 \\ 0 \end{bmatrix} dt + \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ 0 \\ \frac{s\tan\phi}{L} \\ 0 \end{bmatrix} dt + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \left(\begin{bmatrix} a \\ \zeta \end{bmatrix} dt + \begin{bmatrix} \sigma & 0 \\ 0 & \nu \end{bmatrix} dw \right). \quad (44)$$

The control input $u := \begin{bmatrix} a & \zeta \end{bmatrix}^\top$ consists of acceleration a and the front wheel angular rate ζ. σ and ν are the noise level parameters. In the simulation, we set $\sigma = \nu = 0.07$, $k = 0.2$, $t_0 = 0$, $T = 10$, $L = 0.05$, $x_0 = \begin{bmatrix} -0.4 & -0.4 & 0 & 0 & 0 \end{bmatrix}^\top$, $V(x) = p_x^2 + p_y^2$ and $\psi(x(T)) = p_x^2(T) + p_y^2(T)$. We solve the chance-constrained problem (6) via path integral approach using Algorithm 1. Since the state-space of the system is of high order we do not use FDM to solve this problem. For Monte Carlo sampling, 10^4 trajectories and a step size equal to 0.1 are used.

In Figure 9, we plot 100 sample trajectories generated using synthesized optimal policies for two values of Δ. The trajectories are color-coded similar to the previous two examples. We can observe that the failure probability for the lower value of Δ is less as compared to that of the higher value of Δ. Figure 10 represents how the value of η^* and $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,; \eta^*))$ change with respect to Δ. As expected the value of η^* reduces and that of P_{fail} increases as Δ increases.

The algorithm is implemented on the same machine mentioned in the previous two examples. It is run in MATLAB, and the Monte Carlo simulations are run parallelly using MATLAB's parallel processing toolbox. The average number of iterations required by the proposed algorithm to converge is 11. We choose the initial η to be 0.77 and the learning rate γ to be 0.1. The average computation time required per iteration is 12 sec. Note that no special effort was made to optimize the algorithm for speed. We plan to reduce the computation time of the algorithm in future work.

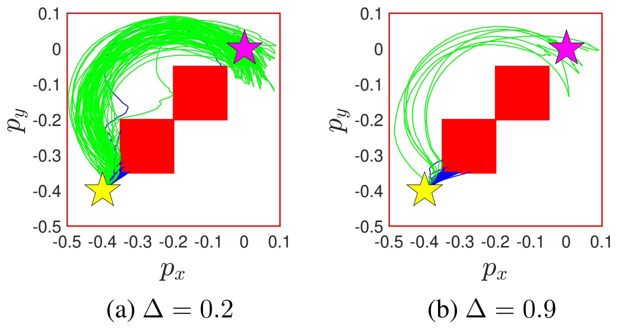


Figure 9: Robot navigation problem for a car model. The start position is shown by a yellow star and the target position (the origin) by a magenta star. 100 sample trajectories generated using optimal control policies for two values of Δ are shown. The trajectories are color-coded; blue paths collide with the obstacle or the outer boundary, while the green paths converge in the neighborhood of the magenta star.

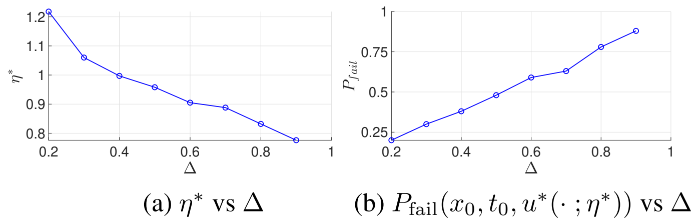


Figure 10: η^* and $P_{\text{fail}}(x_0, t_0, u^*(\cdot\,; \eta^*))$ vs Δ for a car model using path integral control.

2.8 Publications

- **A. Patil**, A. Duarte, A. Smith, F. Bisetti, T. Tanaka, "Chance-Constrained Stochastic Optimal Control via Path Integral and Finite Difference Methods," *2022 IEEE Conference on Decision and Control (CDC)*
- **A. Patil**, A. Duarte, F. Bisetti, T. Tanaka, "Strong Duality and Dual Ascent Approach to Continuous-Time Chance-Constrained Stochastic Optimal Control," *submitted to Transactions on Automatic Control*

2.9 Future Work

In order to prove the strong duality between the primal chance-constrained problem (6) and its dual (14), we required Assumption 3, where we assumed continuity of the function $\eta \mapsto P_{\text{fail}}(x_0, t_0, u^*(x, t; \eta)) : [0, \infty) \to [0, 1]$. We will conduct a formal analysis which will prove that the Assumption 3 holds true under mild conditions. Along with that, we plan to reduce the computation time of the proposed algorithm. We also plan to conduct the sample complexity analysis of the path integral control in order to investigate how the accuracy of Monte Carlo sampling affects the solution of chance-constrained SOC problems. Some preliminary results on this topic are presented in [7, 66]. In order to achieve a strong duality between the primal chance-constrained problem and its dual, in this work, we require a certain assumption on the system dynamics and the cost function (Assumption 1). This assumption restricts the class of applicable system models and cost functions. In future work, we plan to find alternatives in order to get rid of this restrictive assumption (one such solution is provided in [37]). Another topic of future investigation is chance-constrained stochastic games.

3 Two-Player Zero-Sum Stochastic Differential Game

3.1 Motivation and Literature Review

Interactions among multiple agents are prevalent in many fields such as economics, politics, and engineering. Game theory studies the collective decision-making process of multiple interacting agents [67]. Two-person zero-sum games involve two players with conflicting interests, and one player's gain is the other's loss. Pursuit-evasion games (competition between a pursuer and an evader) [68, 69] and robust control (competition between a controller and the nature) [67] are some examples of two-player zero-sum games. In this work, we consider a two-player stochastic differential game (SDG) in which the outcome of the game depends not only on the decision of both players but also on the stochastic input added by nature.

When the game dynamics and cost functions are known, the saddle-point equilibrium of a two-player zero-sum SDG can be characterized by the Hamilton-Jacobi-Isaacs (HJI) partial differential equation (PDE). The analytical solutions of the HJI PDEs are in general not available, and one needs to resort on numerical methods such as grid-based approaches [70], [71] to solve these PDEs approximately. However, the grid-based approaches suffer from *curse of dimensionality*, making them computationally intractable for systems with large dimensions [72]. Moreover, in general, the solutions can not be computed in real-time using these methods; they need to be precomputed as lookup tables and recalled for use in an online setting [71].

Several reinforcement learning algorithms have also been proposed to find approximate solutions to game problems. A reinforcement-learning-based adaptive dynamic programming algorithm is proposed in [73] to determine online a saddle-point solution of linear continuous-time two-player zero-sum differential games. A deep reinforcement learning algorithm based on updating players' policies simultaneously is proposed in [74] to solve two-player zero-sum games.

These methods assume deterministic game dynamics and do not consider system uncertainties. In the presence of system uncertainties, the performance and safety of both players are affected unpredictably, if the uncertainties are not accommodated while designing policies. An effective uncertainty evaluation method, the multivariate probabilistic collocation, was used in [75] with integral reinforcement learning to solve multi-player SDGs for linear system dynamics online. A two-person zero-sum stochastic game with discrete states and actions is solved in [76] using Bayesian inverse reinforcement learning. Common challenges in the learning-based methods include training efficiency, and rigorous theoretical guarantees on convergence and optimality. Moreover, these approaches do not explicitly take into account the players' failure probabilities while synthesizing their policies.

In our work, we formulate a continuous-time, nonlinear, two-player zero-sum SDG on a state space modeled by an Itô stochastic differential equation. Since the stochastic uncertainties in our model are unbounded, both players have nonzero probabilities of failure. Failure occurs when the state of the game enters into predefined undesirable domains, and one player's failure is the other's success. Our objective is to solve a game in which each player seeks to minimize its risk of failure (failure probability)[2] along with its control cost; hence, the name *risk-minimizing zero-sum SDG*. We explain the risk-minimizing zero-sum SDG via the following example:

Example 2. *Consider a pursuit-evasion game in which the pursuer catches the evader if they are less than a certain distance ρ away from each other. In this setting, the evader wishes to minimize its probability of entering the ball of radius ρ centered at the pursuer's location. Whereas, the pursuer wishes to minimize the probability of staying out of the ball of radius ρ centered at the evader's location. The goal of each player is to balance the trade-off between the above probabilities (probabilities of failure) and the control cost (for e.g., their energy consumption). This problem can be formulated as a risk-minimizing zero-sum SDG.*

We derive a sufficient condition for this game to have a saddle-point equilibrium and show that it can be solved via an HJI PDE with *Dirichlet boundary condition*. Under certain assumptions on the system dynamics and cost function, we establish the existence and uniqueness of the saddle-point equilibrium of the formulated risk-minimizing zero-sum SDG. Furthermore, explicit expressions for the saddle-point policies are derived which can be numerically evaluated using path integral control. The use of the path integral technique to solve stochastic games was proposed in [79]. In this work, we generalize their work and develop a path integral formulation to solve HJI PDEs with Dirichlet boundary conditions and find saddle-point equilibria of risk-minimizing zero-sum SDGs. The proposed framework allows us to solve the game online using Monte Carlo simulations of system trajectories, without the need for any offline training or precomputations.

[2]Throughout this section the word "risk" simply means the probability of failure. It is not our intention to discuss various risk measures existing in the literature (e.g. [77, 78]).

3.2 Contributions

The contributions of this work are as follows:

1. We formulate a continuous-time risk-minimizing zero-sum SDG in which players aim at balancing the trade-off between the failure probability and control cost. A sufficient condition for this game to have a saddle-point equilibrium is derived, and it is shown that this game can be solved via an HJI PDE with a Dirichlet boundary condition.

2. Under certain assumptions on the system dynamics and cost function, we establish the existence and uniqueness of the saddle-point solution. We also obtain explicit expressions for the saddle-point policies which can be numerically evaluated using path integral control.

3. The proposed control synthesis framework is validated by applying it to two classes of risk-minimizing zero-sum SDGs, namely a disturbance attenuation problem and a pursuit-evasion game.

Notations

We use the same notations as described in Section 2.

3.3 Problem Formulation

We consider a two-player zero-sum stochastic differential game (SDG) on a finite time horizon $t \in [t_0, T]$, $t_0 < T$. Consider a class of control-affine stochastic systems described by the following :

$$dx(t) = f\left(x(t), t\right) dt + G_u\left(x(t), t\right) u(x(t), t) dt + G_v\left(x(t), t\right) v\left(x(t), t\right) dt + \Sigma\left(x(t), t\right) dw(t) \tag{45}$$

where $x(t) \in \mathbb{R}^n$ is the state, $u(x(t), t) \in \mathbb{R}^m$ is the control input of the first player (henceforth called the agent), and $v(x(t), t) \in \mathbb{R}^l$ that of the second player (called the adversary). $w(t) \in \mathbb{R}^k$ is a k-dimensional standard Wiener process on a suitable probability space $(\Omega, \mathcal{F}, P)$. We assume sufficient regularity in the functions $f\left(x(t), t\right) \in \mathbb{R}^n$, $G_u\left(x(t), t\right) \in \mathbb{R}^{n\times m}$, $G_v\left(x(t), t\right) \in \mathbb{R}^{n \times l}$ and $\Sigma\left(x(t), t\right) \in \mathbb{R}^{n\times k}$ so that a unique strong solution of (45) exists [59, Chapter 1]. Both the control inputs u, and v are assumed to be square integrable (i.e., of finite energy).

Let $\mathcal{X}_s \subseteq \mathbb{R}^n$ be a bounded open set representing a safe region, $\partial\mathcal{X}_s$ be its boundary, and closure $\overline{\mathcal{X}_s} = \mathcal{X}_s \cup \partial\mathcal{X}_s$. Suppose that the agent tries to keep the system (45) in the safe set $\mathcal{X}_s$ for the entire time horizon $[t_0, T]$ of the game, whereas the adversary seeks the opposite. For example, in pursuit-evasion games, the safe set $\mathcal{X}_s$ could be a region outside the ball of radius ρ, centered at the adversary's location. Or, in the disturbance rejection problems, if an agent wishes to navigate through obstacles in the presence of adversarial disturbances, then the region outside obstacles could be considered as a safe set. Suppose, when the game starts at t_0, the system is in the safe set i.e., $x(t_0) = x_0 \in \mathcal{X}_s$. If the system leaves the region $\mathcal{X}_s$ at any time $t \in (t_0, T]$, we say that the agent fails. On the other hand, the adversary fails if the system stays in $\mathcal{X}_s$ for all $t \in [t_0, T]$. Therefore, we define the agent's probability of failure $P_{\text{fail}}^{\text{ag}}$ as

$$P_{\text{fail}}^{\text{ag}} := P_{x_0,t_0}\left(\bigvee_{t\in(t_0,T]} x(t) \notin \mathcal{X}_s\right) \tag{46}$$

and the adversary's probability of failure $P_{\text{fail}}^{\text{ad}} := 1 - P_{\text{fail}}^{\text{ag}}$. We define the terminal time t_f of the game as

$$t_f := \begin{cases} T, & \text{if } \ x(t) \in \mathcal{X}_s, \forall t \in (t_0, T), \\ \inf\left\{t \in (t_0, T) : x(t) \notin \mathcal{X}_s\right\}, & \text{otherwise.} \end{cases}$$

Alternatively, t_f can be defined as

$$t_f := \inf\{t > t_0 : (x(t), t) \notin \mathcal{Q}\} \tag{47}$$

where $\mathcal{Q} = \mathcal{X}_s \times [t_0, T)$ is a bounded set with the boundary $\partial\mathcal{Q} = (\partial\mathcal{X}_s \times [t_0, T]) \cup (\mathcal{X}_s \times \{T\})$, and closure $\overline{\mathcal{Q}} = \mathcal{Q} \cup \partial\mathcal{Q} = \overline{\mathcal{X}_s} \times [t_0, T]$. Note that by the above definitions, $(x(t_f), t_f) \in \partial\mathcal{Q}$ and the agent's failure probability $P_{\text{fail}}^{\text{ag}}$ in (46) can be written in terms of t_f as

$$P_{x_0,t_0}\left(\bigvee_{t\in(t_0,T]} x(t) \notin \mathcal{X}_s\right) = \mathbb{E}_{x_0,t_0}\left[\mathbb{1}_{x(t_f)\in\partial\mathcal{X}_s}\right].$$

Since the stochastic uncertainty of the system (45) is modeled with an unbounded distribution, both the agent and the adversary have nonzero probabilities of failure. In our two-player SDG setting, we assume that both players aim to design an optimal policy against the *worst* possible opponent's policy such that their own *risk of failure* is minimized. Therefore, we define the following *risk-minimizing* cost function:

$$C\left(x_0, t_0; u, v\right) := \eta\, \mathbb{E}_{x_0,t_0}\left[\mathbb{1}_{x(t_f)\in\partial\mathcal{X}_s}\right] + \mathbb{E}_{x_0,t_0}\left[\psi(x(t_f))\cdot\mathbb{1}_{x(t_f)\in\mathcal{X}_s} + \int_{t_0}^{t_f} L(x(t), u(t), v(t), t)dt\right]. \tag{48}$$

The first term indicates the penalty associated with the agent's failure with the weight parameter $\eta > 0$. $\psi\left(x(t_f)\right)$ and $L\left(x(t), u(t), v(t), t\right)$ denote the terminal and running costs, respectively. Note that the game ends at t_f and the system doesn't evolve after that (this is motivated by the applications where "collision" or "capture" ends the game). Therefore, the running cost is integrated over the time horizon $[t_0, t_f]$. The agent tries to minimize C by controlling u, whereas the adversary tries to maximize it by controlling v. The weight parameter η balances the trade-off between the control cost and the failure probability.

Notice that if we define $\phi : \overline{\mathcal{X}_s} \to \mathbb{R}$ as:

$$\phi\left(x\right) := \psi\left(x\right)\cdot \mathbb{1}_{x\in\mathcal{X}_s} + \eta\cdot\mathbb{1}_{x\in\partial\mathcal{X}_s},$$

then, the first term in (48) can be absorbed in a new terminal cost function ϕ as follows:

$$C(x_0, t_0; u, v) = \mathbb{E}_{x_0,t_0}\left[\phi\left(x(t_f)\right) + \int_{t_0}^{t_f} L(x, u, v, t)\, dt\right]. \tag{49}$$

In this work, we consider the following running cost that is quadratic in u and v:

$$L(x, u, v, t) = V(x, t) + \frac{1}{2}u^T R_u(x, t)\, u - \frac{1}{2}v^T R_v(x, t)\, v \tag{50}$$

where $V\left(x, t\right)$ denotes a state dependent cost, and $R_u\left(x, t\right) \in \mathbb{R}^{m\times m}$ and $R_v\left(x, t\right) \in \mathbb{R}^{l\times l}$ are given positive definite matrices (for all values of x and t). Now, we formulate our risk-minimizing zero-sum SDG as follows:

Problem 3 (Risk-Minimizing Zero-Sum SDG)**.**

$$\begin{aligned}
&\min_u \max_v\ \mathbb{E}_{x_0,t_0}\left[\phi\left(x(t_f)\right) + \int_{t_0}^{t_f}\left(\frac{1}{2}u^\top R_u u - \frac{1}{2}v^\top R_v v + V\right)dt\right]\\
&\textit{s.t.}\ \ dx = f dt + G_u u dt + G_v v dt + \Sigma dw,\\
&\qquad\quad x(t_0) = x_0.
\end{aligned} \tag{51}$$

where the admissible policies u, v are measurable with respect to the σ-algebra generated by $x(s), t_0 \le s \le t$.

Note that Problem 3 is a *variable-terminal-time* zero-sum SDG where the terminal time is determined by (47).

3.4 Synthesis of Optimal Control Policies

This section presents the main results. In Section 3.4.1, we show that Problem 3 can be solved using backward dynamic programming which results into an HJI PDE with a Dirichlet boundary condition. In Section 3.4.2, we find a solution of a class of risk-minimizing zero-sum SDGs via path integral control.

3.4.1 Backward Dynamic Programming

Notice that the cost function of the risk-minimizing zero-sum SDG (51) possesses the time-additive Bellman structure. Therefore, Problem 3 can be solved by utilizing the principle of dynamic programming. For each $(x, t) \in \overline{\mathcal{Q}}$, and admissible policies u, v over $[t, T)$, define the cost-to-go function:

$$C\left(x, t; u, v\right) = \mathbb{E}_{x,t}\left[\phi\left(x(t_f)\right)\right] + \mathbb{E}_{x,t}\left[\int_t^{t_f}\left(\frac{1}{2}u^\top R_u u - \frac{1}{2}v^\top R_v v + V\right)dt\right].$$

Definition 4 (Saddle-point solution)**.** *[80, Chapter 2]: Given a two-player zero-sum differential game, a pair of admissible policies (u^*, v^*) over $[t, T)$ constitutes a saddle-point solution, if for each $(x, t) \in \overline{\mathcal{Q}}$, and admissible policies (u, v) over $[t, T)$,*

$$C(x, t; u^*, v) \leq C^* := C(x, t; u^*, v^*) \leq C(x, t; u, v^*).$$

The quantity C^* is the value of the game. The value of the game is defined if it satisfies the following relation

$$C^* = \min_u \max_v C(x, t; u, v) = \max_v \min_u C(x, t; u, v).$$

The following theorem provides the sufficient condition for a saddle-point solution of Problem 3 to exist.

Theorem 7. *Suppose there exists a function $J : \overline{\mathcal{Q}} \to \mathbb{R}$ such that*

(a) $J(x, t)$ is continuously differentiable in t and twice continuously differentiable in x in the domain $\mathcal{Q}$;

(b) $J(x, t)$ solves the following stochastic HJI PDE:

$$\begin{cases} -\partial_t J = V + f^\top \partial_x J + \frac{1}{2} Tr\left(\Sigma\Sigma^\top \partial_x^2 J\right) + \frac{1}{2}(\partial_x J)^\top \left(G_v R_v^{-1} G_v^\top - G_u R_u^{-1} G_u^\top\right)\partial_x J, & \forall (x,t) \in \mathcal{Q}, \\ \lim\limits_{\substack{(x,t)\to(y,s) \\ (x,t)\in\mathcal{Q}}} J(x, t) = \phi(y), & \forall (y, s) \in \partial\mathcal{Q}. \end{cases} \tag{52}$$

Then, the following statements hold:

(i) $J(x, t)$ is the value of the game formulated in Problem 3. That is,

$$J(x, t) = \min_u \max_v C(x, t; u, v) = \max_v \min_u C(x, t; u, v), \quad \forall\, (x, t) \in \overline{\mathcal{Q}}.$$

(ii) The optimal solution to Problem 3 is given by

$$u^*(x, t) = -R_u^{-1}(x, t)\, G_u{}^\top(x, t)\, \partial_x J(x, t), \tag{53}$$

$$v^*(x, t) = R_v^{-1}(x, t)\, G_v{}^\top(x, t)\, \partial_x J(x, t). \tag{54}$$

Proof. Let $J(x, t)$ be the function satisfying (a) and (b). By Dynkin's formula [59, 1], for each $(x, t) \in \overline{\mathcal{Q}}$ we have

$$\mathbb{E}_{x,t}\left[J\left(x(t_f), t_f\right)\right] = J(x, t) + \mathbb{E}_{x,t}\left[\int_t^{t_f} \left(\partial_t J + (f + G_u u + G_v v)^T (\partial_x J) + \frac{1}{2}\text{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right)\right) ds\right]. \tag{55}$$

By the boundary condition of the PDE (52),
$J\left(x(t_f), t_f\right) = \phi\left(x(t_f)\right)$. Hence, from (55), we obtain

$$J(x, t) = \mathbb{E}_{x,t}\left[\phi\left(x(t_f)\right)\right] - \mathbb{E}_{x,t}\left[\int_t^{t_f} \left(\partial_t J + (f + G_u u + G_v v)^T (\partial_x J) + \frac{1}{2}\text{Tr}\left(\Sigma\Sigma^T \partial_x^2 J\right)\right) ds\right].$$

Now, notice that the right-hand side of the PDE in (52) can be expressed as the minimum and maximum value of a quadratic form in u and v, respectively, as follows:

$$-\partial_t J = \min_u \max_v \left[\frac{1}{2} u^\top R_u u - \frac{1}{2} v^\top R_v v + V + (f + G_u u + G_v v)^\top \partial_x J + \frac{1}{2}\text{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right)\right]. \tag{56}$$

Observe that the "$\min\limits_u$", "$\max\limits_v$" operations in (56) can be interchanged. Hence, the game formulated in (51) satisfies the *Isaacs condition* [81]. If $\hat{u}$ and $\hat{v}$ represent the minimum and maximum values of the right-hand side of (56) respectively, then

$$\hat{u} = -R_u^{-1} G_u^\top \partial_x J, \qquad \hat{v} = R_v^{-1} G_v^\top \partial_x J. \tag{57}$$

Therefore, for an arbitrary u, we have

$$-\partial_t J \leq \left[\frac{1}{2} u^\top R_u u - \frac{1}{2} \hat{v}^\top R_v \hat{v} + V + (f + G_u u + G_v \hat{v})^\top \partial_x J + \frac{1}{2}\text{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right)\right]. \tag{58}$$

Now, notice that the equality in (56) holds for any v. Replacing v by $\hat{v}$ in (56) yields

$$J(x,t) = \mathbb{E}_{x,t}\left[\phi\left(x(t_f)\right)\right] - \mathbb{E}_{x,t}\left[\int_t^{t_f}\left(\partial_t J + (f + G_u u + G_v \hat{v})^T(\partial_x J) + \frac{1}{2}\text{Tr}\left(\Sigma\Sigma^\top \partial_x^2 J\right)\right)ds\right]. \tag{59}$$

Combining (58) and (59), we obtain

$$\begin{aligned} J(x,t) &\leq \mathbb{E}_{x,t}\left[\phi(x(t_f)) + \int_t^{t_f}\left(\frac{1}{2}u^T R_u u - \frac{1}{2}\hat{v}^T R_v \hat{v} + V\right)ds\right] \\ &= C\left(x,t;u,\hat{v}\right) \end{aligned}$$

where the equality holds iff $\hat{u} = -R_u^{-1}G_u^\top \partial_x J$. Similarly, for an arbitrary v, we can show that

$$J(x,t) \geq C\left(x,t;\hat{u},v\right) \tag{60}$$

where the equality holds iff $\hat{v} = R_v^{-1}G_v^\top \partial_x J$. Therefore, from Definition 4, it follows that the pair of policies $(\hat{u},\hat{v})$ defined in (57) provides the optimal solution to the zero-sum game formulated in Problem 3 and $J(x,t)$ is the value of the game. □

Remark 3. *Theorem 7 does not say anything about the existence of a function $J(x,t)$ satisfying statements (a) and (b), and it is not in the scope of this proposal. However, in Section 3.4.2, we focus on a special case in which (52) can be linearized where the existence and uniqueness of such a function is guaranteed.*

3.4.2 Path Integral Solution

In this section, we derive a path integral formulation to solve a class of risk-minimizing zero-sum SDGs that satisfy certain assumptions on the system dynamics and cost function. Let $\xi(x,t)$ be the logarithmic transformation of the value function $J(x,t)$ defined as

$$J(x,t) = -\lambda \log\left(\xi\left(x,t\right)\right) \tag{61}$$

where λ is a proportionality constant to be defined. Applying the transformation in (61) to (52) yields

$$\begin{cases} \partial_t \xi = \dfrac{V\xi}{\lambda} - \dfrac{1}{2}\text{Tr}\left(\Sigma\Sigma^\top \partial_x^2 \xi\right) + \dfrac{1}{2\xi}(\partial_x \xi)^T \Sigma\Sigma^\top \partial_x \xi + \dfrac{\lambda}{2\xi}(\partial_x \xi)^\top\left(G_v R_v^{-1} G_v^\top - G_u R_u^{-1} G_u^\top\right)\partial_x \xi - f^\top \partial_x \xi, & \forall (x,t) \in \mathcal{Q}, \\ \displaystyle\lim_{\substack{(x,t)\to(y,s)\\(x,t)\in\mathcal{Q}}} \xi(x,t) = \exp\left(-\tfrac{\phi(y)}{\lambda}\right), & \forall (y,s) \in \partial\mathcal{Q}. \end{cases} \tag{62}$$

Now, we make the following assumption:

Assumption 4. *For all $(x,t) \in \overline{\mathcal{Q}}$, there exists a constant $\lambda > 0$ such that*

$$\Sigma(x,t)\Sigma^\top(x,t) = \lambda G_u(x,t)R_u^{-1}(x,t)G_u^\top(x,t) - \lambda G_v(x,t)R_v^{-1}(x,t)G_v^\top(x,t). \tag{63}$$

Assumption 4 is similar to the assumption required in the path integral formulation of a single agent stochastic control problem (25). A possible interpretation of condition (63) is that in a direction with high noise variance, the agent's control cost has to be low whereas that of the adversary has to be high. Therefore, the weights of the control cost R_u and R_v need to be tuned appropriately for the given diffusion coefficient $\Sigma(x,t)$ and the control gains $G_u(x,t)$ and $G_v(x,t)$ in the system dynamics (45). Assumption 4 also implies that the stochastic noise has to enter the system dynamics via the control channels. Therefore, in what follows, we assume that system (45) can be partitioned into subsystems that are directly and non-directly driven by the noise as:

$$\begin{bmatrix} dx^{(1)} \\ dx^{(2)} \end{bmatrix} = \begin{bmatrix} f^{(1)}(x,t) \\ f^{(2)}(x,t) \end{bmatrix} dt + \begin{bmatrix} \mathbf{0} \\ G_u^{(2)}(x,t) \end{bmatrix} u(x,t)dt + \begin{bmatrix} \mathbf{0} \\ G_v^{(2)}(x,t) \end{bmatrix} v(x,t)dt + \begin{bmatrix} \mathbf{0} \\ \Sigma^{(2)}(x,t) \end{bmatrix} dw \tag{64}$$

where $\mathbf{0}$ denotes a zero matrix of appropriate dimensions. By assuming a λ satisfying Assumption 4 holds in (62), we obtain the linear PDE in ξ with Dirichlet boundary condition:

$$\begin{cases} \partial_t \xi = \frac{V\xi}{\lambda} - f^\top \partial_x \xi - \frac{1}{2}\text{Tr}\left(\Sigma\Sigma^\top \partial_x^2 \xi\right), & \forall (x,t) \in \mathcal{Q}, \\ \displaystyle\lim_{\substack{(x,t)\to(y,s)\\(x,t)\in\mathcal{Q}}} \xi(x,t) = \exp\left(-\tfrac{\phi(y)}{\lambda}\right), & \forall (y,s) \in \partial\mathcal{Q}. \end{cases} \tag{65}$$

The solution of a linear Dirichlet boundary value problem of the form (65) exits under a sufficiently regular boundary condition, and it is unique [61, Chapter 6]. Furthermore, the solution admits the Feynman-Kac representation [82]. Suppose $\hat{x}(t) \in \mathbb{R}^n$ is an uncontrolled process driven by the following SDE:

$$d\hat{x}(t) = f(\hat{x}(t),t)dt + \Sigma(\hat{x}(t),t)dw(t) \tag{66}$$

and let $\hat{t}_f \coloneqq \inf\{t > t_0 : (\hat{x}(t), t) \notin \mathcal{Q}\}$. Then, the solution of the PDE (65) is given as

$$\xi\left(x,t\right) = \mathbb{E}_{x,t}\left[\exp\left(-\frac{1}{\lambda}S\left(\tau\right)\right)\right] \tag{67}$$

where $S\left(\tau\right)$ denotes the cost-to-go of a trajectory τ of the uncontrolled system (66) starting at (x,t):

$$S\left(\tau\right) = \phi\left(\hat{x}(\hat{t}_f)\right) + \int_t^{\hat{t}_f} V\left(\hat{x}(t), t\right) dt. \tag{68}$$

Equation (67) provides a path integral form for the exponentiated value function $\xi\left(x,t\right)$, which can be numerically evaluated using Monte Carlo sampling of trajectories generated by the uncontrolled SDE (66).

We now obtain the expressions for the saddle-point policies via the following theorem:

Theorem 8. *Suppose Assumption 4 holds and the system (45) can be partitioned as (64). Then, a saddle-point solution of the risk-minimizing zero-sum SDG (51) exists, is unique, and is given by*

$$u^*(x,t)dt = \mathcal{G}_u(x,t)\frac{\mathbb{E}_{x,t}\left[exp\left(-\frac{1}{\lambda}S\left(\tau\right)\right)\Sigma^{(2)}(x,t)\,dw\right]}{\mathbb{E}_{x,t}\left[exp\left(-\frac{1}{\lambda}S\left(\tau\right)\right)\right]}, \tag{69}$$

where

$$\mathcal{G}_u = R_u^{-1}G_u^{(2)\top}\left(G_u^{(2)}R_u^{-1}G_u^{(2)\top} - G_v^{(2)}R_v^{-1}G_v^{(2)\top}\right)^{-1}$$

and

$$v^*(x,t)dt = \mathcal{G}_v\left(x,t\right)\frac{\mathbb{E}_{x,t}\left[exp\left(-\frac{1}{\lambda}S\left(\tau\right)\right)\Sigma^{(2)}(x,t)\,dw\right]}{\mathbb{E}_{x,t}\left[exp\left(-\frac{1}{\lambda}S\left(\tau\right)\right)\right]}, \tag{70}$$

where

$$\mathcal{G}_v = -R_v^{-1}G_v^{(2)\top}\left(G_u^{(2)}R_u^{-1}G_u^{(2)\top} - G_v^{(2)}R_v^{-1}G_v^{(2)\top}\right)^{-1}.$$

Proof. The existence and uniqueness of the saddle-point solution follow from the existence and uniqueness of the linear Dirichlet boundary value problem (65) [61, Chapter 6] and from Theorem 7. The saddle-point solution $u^*(x,t)$ (53) and $v^*(x,t)$ (54) can be computed by taking the gradient of (67) with respect to x and using the condition (63). (The derivation of (53) and (54) is in the same vein as the derivation of optimal controls in single-agent settings [37, 3]; not presented here for brevity.) □

Equations (69) and (70) provide the path integral forms for the saddle-point equilibrium. Similar to (67), the expectations in (69) and (70) can be numerically evaluated in real-time via the Monte Carlo sampling of the trajectories generated by the uncontrolled SDE (66). Path integral framework allows us to solve the game online without requiring any offline training or precomputations. Even though Monte Carlo simulations must be performed in real-time in order to evaluate (69, 70) for the current (x,t), these simulations can be massively parallelized through the use of GPUs.

3.5 Simulation Results

In this section, we apply the path integral framework to two classes of risk-minimizing zero-sum SDGs (51): a disturbance attenuation problem and a pursuit-evasion game.

3.5.1 Disturbance Attenuation Problem

Consider a special class of systems (1):

$$dx = f(x,t)dt + G_u(x,t)\Big(u(x,t)dt + v\,(x,t)\,dt + dw\Big) \tag{71}$$

where $u\,(x,t) \in \mathbb{R}^m$ is the control input, $v\,(x,t) \in \mathbb{R}^m$ is the bounded disturbance and $w(t) \in \mathbb{R}^m$ is a Wiener process. Here, we have two sources of noise that corrupt the system's control input u: the bounded noise v whose statistics are unknown, and the white noise dw. In the disturbance attenuation problem, the objective is to design a policy u in the presence of stochastic noise and bounded disturbance v such that the system's control performance $\mathbb{E}_{x_0,t_0}\Big[\phi\,(x(t_f)) + \int_{t_0}^{t_f}\big(\frac{1}{2}u^\top u + V\big)\,dt\Big]$ is minimized. This problem can be solved using the following zero-sum SDG, where u is considered as a control input of the first player (agent) and v that of the second player (adversary):

$$\min_u \max_v \mathbb{E}_{x_0,t_0}\Bigg[\phi\,(x(t_f)) + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top u - \frac{\gamma^2}{2}v^T v + V\Big)\,dt\Bigg]. \tag{72}$$

γ is a given positive constant that determines the level of disturbance attenuation. Theorem 9 provides an upper bound on the system's control performance (in the presence of a bounded disturbance v) that can be obtained by solving the game (72).

Theorem 9. *Suppose (u_γ^*, v_γ^*) represent the saddle-point policies of the SDG (72) for any γ, and let*

$$\delta_\gamma \coloneqq \mathbb{E}_{x_0,t_0}^{u_\gamma^*,v_\gamma^*}\left[\int_{t_0}^{t_f} v^\top v\; dt\right] \tag{73}$$

where the superscript on $\mathbb{E}$ denotes the policies under which the expectation is computed. Then, for all adversarial policies v such that $\mathbb{E}_{x_0,t_0}^{u_\gamma^*,v}\left[\int_{t_0}^{t_f} v^\top v\; dt\right] \le \delta$ (for any $\delta > 0$), we get the following upper bound on the system's control performance in the presence of disturbance v:

$$\mathbb{E}_{x_0,t_0}^{u_\gamma^*,v_\gamma^*}\Bigg[\phi\,(x(t_f)) + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top u + V\Big)dt\Bigg] + \frac{\gamma^2}{2}\,(\delta - \delta_\gamma) \ge \mathbb{E}_{x_0,t_0}^{u_\gamma^*,v}\Bigg[\phi\,(x(t_f)) + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top u + V\Big)dt\Bigg]. \tag{74}$$

Proof. Consider cost of the SDG (72) under the saddle-point policies (u_γ^*, v_γ^*):

$$\mathbb{E}_{x_0,t_0}^{u_\gamma^*,v_\gamma^*}\Bigg[\phi\,(x(t_f)) + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top u + V - \frac{\gamma^2}{2}v^\top v\Big)\,dt\Bigg] = \mathbb{E}_{x_0,t_0}^{u_\gamma^*,v_\gamma^*}\Bigg[\phi\,(x(t_f)) + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top u + V\Big)\,dt\Bigg] - \frac{\gamma^2}{2}\delta_\gamma \tag{75a}$$

$$\ge \mathbb{E}_{x_0,t_0}^{u_\gamma^*,v}\Bigg[\phi\,(x(t_f)) + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top u + V - \frac{\gamma^2}{2}v^\top v\Big)\,dt\Bigg] \tag{75b}$$

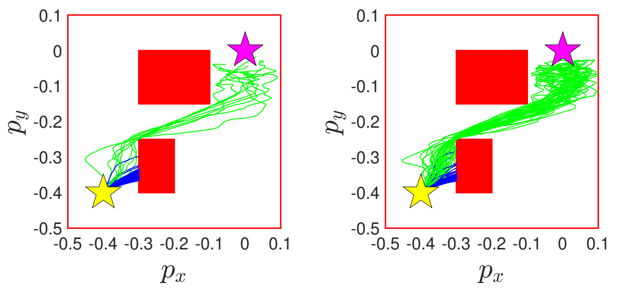
$$\ge \mathbb{E}_{x_0,t_0}^{u_\gamma^*,v}\Bigg[\phi\,(x(t_f)) + \int_{t_0}^{t_f}\Big(\frac{1}{2}u^\top u + V\Big)\,dt\Bigg] - \frac{\gamma^2}{2}\delta. \tag{75c}$$

The equation (75a) follows from (73). For any adversarial policy v, the inequality (75b) follows because v_γ^* maximizes the cost in (72). The inequality (75c) follows from the bound on $\mathbb{E}_{x_0,t_0}^{u_\gamma^*,v}\left[\int_{t_0}^{t_f} v^\top v\; dt\right]$. Using (75a) and (75c), we get the desired inequality (74). □

In order to solve the HJI PDE associated with the game (72) via the path integral framework described in Section 3.4.2, it is necessary to find a constant $\lambda > 0$ (by Assumption 4) such that

$$\lambda\left(1 - \frac{1}{\gamma^2}\right) = 1.$$

Therefore, for all $\gamma > 1$, Assumption 1 is satisfied and as a consequence, the zero-sum SDG (72) admits a unique saddle-point solution.



(a) $\gamma^2 = 2$, $P_{\text{fail}}^{\text{ag}} = 0.9$ (b) $\gamma^2 = 7$, $P_{\text{fail}}^{\text{ag}} = 0.64$

Figure 11: Unicycle navigation in the presence of bounded and stochastic disturbances. The start position is shown by a yellow star and the target position (the origin) by a magenta star. 100 sample trajectories generated using saddle-point policies (u^*, v^*) for two values of γ are shown. The trajectories are color-coded; blue paths collide with the red obstacles or the outer boundary, while the green paths converge in the neighborhood of the magenta star. The failure probabilities of the agent $P_{\text{fail}}^{\text{ag}}$ are noted below each case.

We now present a simulation study of the disturbance attenuation problem using a unicycle navigation example. Consider the following unicycle dynamics model:

$$\begin{bmatrix} dp_x \\ dp_y \\ ds \\ d\theta \end{bmatrix} = -k \begin{bmatrix} p_x \\ p_y \\ s \\ \theta \end{bmatrix} dt + \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ 0 \\ 0 \end{bmatrix} dt + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \left(\begin{bmatrix} a \\ \omega \end{bmatrix} dt + \begin{bmatrix} \Delta a \\ \Delta\omega \end{bmatrix} dt + \begin{bmatrix} \sigma & 0 \\ 0 & \nu \end{bmatrix} dw \right),$$

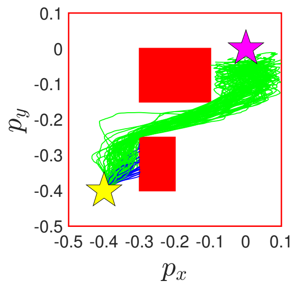
where $(p_x,\ p_y)$, s and θ denote the position, speed, and the heading angle of the unicycle, respectively. The control input $u := \begin{bmatrix} a & \omega \end{bmatrix}^\top$ consists of acceleration a and angular speed ω. $v := \begin{bmatrix} \Delta a & \Delta\omega \end{bmatrix}^\top$ is the bounded disturbance acting on the system's control input, and $dw \in \mathbb{R}^2$ is the white noise with σ and ν being the noise level parameters. As illustrated in Figure 11, the unicycle aims to navigate in a two-dimensional space from its initial position (represented by the yellow star) to the origin (represented by the magenta star), in finite time, while avoiding the red obstacles and the outer boundary. The white region that lies between the outer boundary and the obstacles is the safe region $\mathcal{X}_s$. This is a disturbance attenuation problem since the unicycle aims to design its control policy u in order to minimize the control performance and risk of failure (collision with the obstacles or the outer boundary) under worst-case disturbance v. Therefore, we can formulate this problem as the risk-minimizing zero-sum SDG (72). In the simulation, we set $\sigma = \nu = 0.1$, $k = 0.2$, $t_0 = 0$, $T = 10$, $x_0 = \begin{bmatrix} -0.4 & -0.4 & 0 & 0 \end{bmatrix}^\top$, $V(x) = p_x^2 + p_y^2$ and $\psi(x(T)) = p_x^2(T) + p_y^2(T)$. In order to evaluate the optimal policies (69) and (70) via Monte Carlo sampling, 10^4 trajectories and a step size equal to 0.01 are used. We demonstrate two experiments.

Experiment 1

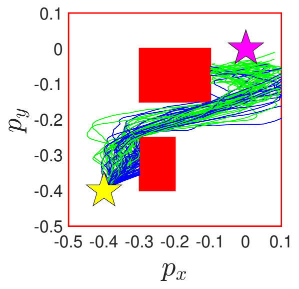
In this experiment, we set $\eta = 0.67$ and plot in Figure 11 100 sample trajectories generated using synthesized saddle-point policies (u^*, v^*) for two values of γ. The trajectories are color-coded; the blue paths collide with the obstacles, while the green paths converge in the neighborhood of the origin (the target position). The figure shows that for a higher value of γ i.e. when the adversary becomes less powerful, the failure probability of the agent $P_{\text{fail}}^{\text{ag}}$ reduces.

Experiment 2

In this experiment, we set $\gamma^2 = 3$, $\eta = 1$, and study the effect of ignoring the adversary. First, we compute saddle-point policies (u^*, v^*) for the game (72) same as Experiment 1 and plot in Figure 12-(a) 100 sample trajectories generated using (u^*, v^*). In this case, the agent is aware of the adversary and designs its policy u^* cautiously. The probability of failure is 23%. In the second case, the agent is not aware of the presence of the adversary and computes its policy (say) $\widetilde{u}^*$ by solving a single agent optimization problem. However, in reality, the adversary is present and suppose it follows the policy v^*. Figure 12-(b) shows 100 sample trajectories generated using $(\widetilde{u}^*, v^*)$. In this case, the agent's performance is poor, it fails 65% of the time. The color-coding of the trajectories is the same as in Experiment 1.



(a) Agent is aware of the adversary, $P_{\text{fail}}^{\text{ag}} = 0.23$



(b) Agent is not aware of the adversary, $P_{\text{fail}}^{\text{ag}} = 0.65$

Figure 12: Unicycle navigation in the presence of bounded and stochastic disturbances. (a) The agent is aware of the presence of an adversary. 100 sample trajectories generated using saddle-point policies (u^*, v^*). (b) The agent is not aware of the presence of an adversary. 100 sample trajectories generated using $(\widetilde{u}^*, v^*)$. The failure probabilities of the agent $P_{\text{fail}}^{\text{ag}}$ are noted for each case.

3.5.2 Pursuit-Evasion Game

Consider a two-player zero-sum SDG on a finite time horizon $[t_0, T]$, in which the adversary is chasing the agent and the agent is trying to escape from the adversary. We will call the adversary as a pursuer and the agent as an evader. Suppose the evader and the pursuer are moving in a two-dimensional plane according to

$$dp_x^E = u_x dt + \sigma_x^E dw_x^E, \qquad dp_x^P = v_x dt + \sigma_x^P dw_x^P, \qquad dp_y^E = u_y dt + \sigma_y^E dw_y^E, \qquad dp_y^P = v_y dt + \sigma_y^P dw_y^P, \quad (76)$$

where $x_E := \begin{bmatrix} p_x^E & p_y^E \end{bmatrix}^\top$ is the position and $u := \begin{bmatrix} u_x & u_y \end{bmatrix}^\top$ is the control input of the evader. Similarly, $x_P := \begin{bmatrix} p_x^P & p_y^P \end{bmatrix}^\top$ and $v := \begin{bmatrix} v_x & v_y \end{bmatrix}^\top$ are the position and control input of the pursuer. $w_x^E, w_y^E, w_x^P, w_y^P$ are independent one-dimensional standard Brownian motions. If at any time $t \in (t_0, T]$, the pursuer gets within a distance ρ of the evader, then it catches the evader and the evader fails. On the other hand, if the evader avoids getting within a distance ρ of the pursuer for the entire time horizon $[t_0, T]$, then that's a failure for the pursuer. The pursuer aims at designing its control policy v in order to maximize the probability of catching the evader, whereas the evader seeks the opposite by designing u. For this two-player differential game, it is the relative position of the pursuer and evader that is important (and relevant), rather than their absolute positions. Let $x := \begin{bmatrix} p_x & p_y \end{bmatrix}$ be the evader's position with respect to the pursuer where

$$p_x = p_x^E - p_x^P, \quad p_y = p_y^E - p_y^P$$

and the origin coincides with the pursuer's position. Thus, the coordinate system is attached to the pursuer and is not fixed in space. The system x follows the SDE

$$dx = \begin{bmatrix} dp_x \\ dp_y \end{bmatrix} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} dt - \begin{bmatrix} v_x \\ v_y \end{bmatrix} dt + \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix} dw \quad (77)$$

where $\sigma_x = \sqrt{(\sigma_x^E)^2 + (\sigma_x^P)^2}$, $\sigma_y = \sqrt{(\sigma_y^E)^2 + (\sigma_y^P)^2}$ and w is a two-dimensional standard Brownian motion. In this game, the safe set $\mathcal{X}_s$ can be defined as $\mathcal{X}_s := \{x \in \mathbb{R}^2 : \|x\| > \rho\}$. Suppose the control cost matrix R_u of the evader is unity and that of the pursuer $R_v = {r_v}^2$, where r_v is a given positive scalar constant. Therefore, the risk-minimizing zero-sum SDG takes the form:

$$\min_u \max_v \mathbb{E}_{x_0,t_0}\left[\phi\left(x(t_f)\right) + \int_{t_0}^{t_f} \left(\frac{1}{2} u^T u - \frac{r_v^2}{2} v^T v + V\right) dt\right]. \quad (78)$$

In order to solve the associated HJI equation of this game via the path integral framework, it is necessary to find a constant $\lambda > 0$ (by Assumption 1) such that

$$\lambda\left(1 - \frac{1}{{r_v}^2}\right) = 1.$$

Therefore, for all $r_v > 1$, Assumption 1 is satisfied and as a consequence, the zero-sum SDG (78) admits a unique saddle-point solution.

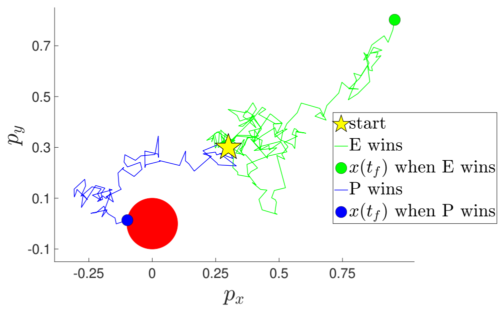


Figure 13: Two sample trajectories of the relative position of the players in a pursuit-evasion game. The start position of the trajectories is shown by a yellow star. The red disc of radius $\rho = 0.1$, centered at the origin represents that the pursuer is within the distance ρ of the evader. The green trajectory never enters the red disc in the horizon $[t_0, T]$, thus, it represents a case when the evader wins. The blue trajectory enters the red disc and thus represents a case when the pursuer wins.

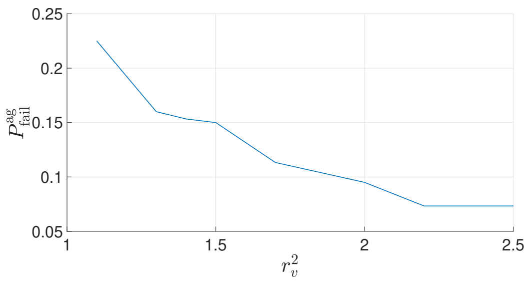


Figure 14: Failure probabilities of the agent (i.e., evader) as a function of r_v, when the players follow the saddle-point policies (u^*, v^*).

In the simulation, we set $\sigma_x^E = \sigma_y^E = \sigma_x^P = \sigma_y^P = \sqrt{0.1}$, $\rho = 0.1$, $t_0 = 0$, $T = 2$, $x_0 = \begin{bmatrix} 0.3 & 0.3 \end{bmatrix}^\top$, $V(x) = \psi\left(x(T)\right) = 0$, $\eta = 0.2$, ${r_v}^2 = 2$. Figure 13 shows a plot of two sample trajectories of the system (77) generated using synthesized saddle-point policies (u^*, v^*). The trajectories start from x_0 shown by the yellow star. The red disc of radius $\rho = 0.1$, centered at the origin represents that the pursuer is within a distance ρ of the evader. The green trajectory never enters the red disc in the horizon $[t_0, T]$, thus, it represents a case when the evader escapes from the pursuer. The blue trajectory, on the other hand, enters the red disc and thus represents a case when the pursuer catches the evader. Figure 14 shows a plot of failure probabilities of the agent (i.e., evader) as a function of r_v, when the players follow the saddle-point policies (u^*, v^*). These values are computed using naïve Monte Carlo sampling, with 400 sample trajectories. The plot shows that as the control cost weight r_v of the adversary (i.e., pursuer) increases, the chances of the evader getting caught reduces.

The aim of the presented simulation studies is to validate the proposed theoretical formulation of the risk-minimizing zero-sum SDGs. Future work will emphasize scaling this framework to higher dimensional and more complex game dynamics.

3.6 Publications

- **A. Patil**, Y. Zhou, D. Fridovich-Keil, T. Tanaka, “Risk-Minimizing Two-Player Zero-Sum Stochastic Differential Game via Path Integral Control,” *2023 IEEE Conference on Decision and Control (CDC)*

3.7 Future Work

In the future, we plan to conduct sample complexity analysis for path integral control in order to investigate how the accuracy of Monte Carlo sampling affects the solution of SDGs. The central challenge in using the path integral framework is the particular requirement on the relationship between the cost function and the noise covariance. This requirement restricts the class of applicable system models and cost functions. In future work, we plan to find alternatives in order to get rid of this restrictive requirement (one such solution is provided in [37]). Another topic of future investigation could be chance-constrained stochastic games in which each player would aim to satisfy a hard bound on its failure probability.

4 Task Hierarchical Control

4.1 Motivation and Literature Review

Robotic systems with a large number of degrees of freedom offer significant versatility; however, this also introduces redundancies. These systems are often used to accomplish multiple subtasks with varying levels of importance, allowing for the establishment of a task hierarchy. One of the most frequently applied methods to accomplish task hierarchical control is the null-space projection technique [83, 84, 85]. In the null-space projection, the top priority task is executed by employing all the capabilities of the system. The second priority task is then applied to the null space of the top priority task. In other words, the task on the second level is executed as well as possible without disturbing or interfering with the first level. The task on level three is then executed without disturbing the two higher-priority tasks, and so forth. The null-space projection technique is based on a hierarchical arrangement of the involved tasks and can be interpreted as instantaneous local optimization.

The null-space projection technique has been widely applied, such as in multi-robot team control [86], whole-body behavior synthesis [87], and manipulator control [88]. A comprehensive overview and comparison of null-space projections is given in [88]. In the literature [83, 88], the individual controllers for the tasks in the hierarchy are designed using simple low-level controllers such as proportional-integral-derivative (PID) controllers. While these controllers are easy to design and are capable of generating control inputs in real time, there is no systematic way to optimize the overall performance of the hierarchy of controllers. Global optimization techniques such as dynamic programming on the other hand minimize some performance index across a whole trajectory. For example, to design optimal control policies for a linear system with a quadratic cost function, the linear quadratic regulator (LQR) is widely used. For nonlinear systems and cost functions, approaches such as iterative LQR (ILQR), and differential dynamic programming (DDP) can be utilized. However, even though global optimization solutions perform better than local optimization solutions, they are impractical for online feedback control, due to the heavy computational requirements. In this work, we employ the path integral control method, a stochastic optimal control framework that can be applied to nonlinear systems and enables the computation of optimal control inputs in real time through Monte Carlo simulations.

Based on the foundational work of [2, 3], the path integral method can be defined as a sampling-based algorithm to compute control input at each time step from a large number of Monte-Carlo simulations of the *uncontrolled dynamics*. Unlike traditional optimal control methods, the path integral approach can directly deal with stochasticity and nonlinearity [82], [89, 90]. Moreover, unlike dynamic programming, it can evaluate control input without solving a high-dimensional Hamilton-Jacobi-Bellman partial differential equation [82]. The Monte Carlo simulations can also be highly parallelized, leveraging the GPU resources available on modern robotics platforms [6], making it particularly effective for real-time control applications.

This work integrates the path integral control approach with the null-space projection technique to overcome their individual limitations and enhance their respective strengths. We explain our idea via the following example:

Example 3. *Consider a platoon of robots navigating in an obstacle-filled environment, tasked with three goals in descending order of importance:*

1. *Avoiding collisions with obstacles*
2. *Steering the platoon's centroid toward a goal position*
3. *Maintaining specific distances between the robots*

Designing an optimal controller using only the path integral method for each robot in the platoon would present scalability challenges due to the method's sampling-based nature. On the other hand, simple low-level controllers (such as PID), while computationally efficient, are difficult to tune manually for a better performance. We propose using local controllers for tasks 1 and 3 while applying the path integral controller to the more complex task 2. In this way, the path integral controller optimizes the centroid's movement, while simpler tasks are handled by local controllers.

In [86], the authors address the issue of the task hierarchical controller falling into local minima by complementing the low-level controllers with the A* search algorithm. In graph-based methods like A*, the optimal trajectory is generated in the workspace, requiring an additional tracking controller to be designed separately, taking into account the system's dynamics. In contrast, the path integral controller computes the optimal policy directly, eliminating the need for such a decoupled approach. Furthermore, path integral controllers can adapt in real time making them particularly suitable for dynamic environments. Graph-based algorithms like A* are primarily designed for static environments and are less effective in dynamic scenarios, as they require re-planning the trajectory each time the environment changes. Although there are graph-based algorithms, such as RRTX, capable of handling dynamic environments [91], these approaches tend to have higher computational costs compared to static planners and often require substantial memory resources as the number of samples increases.

4.2 Contributions

The contributions of this work are as follows:

1. We introduce a new framework for hierarchical task control that combines the null-space projection technique with the path integral control method. This leverages Monte Carlo simulations for real-time computation of optimal control inputs, allowing for the seamless integration of simpler PID-like controllers with a more sophisticated optimal control technique.

2. Despite the wide applicability of the path integral approach, it has not been utilized for solving the task hierarchical control problem to the best of the authors' knowledge. This expands the applicability of path integral control to multi-task robotic systems, enabling a more robust handling of task prioritization.

3. Our simulation studies demonstrate the effectiveness of the proposed approach, showing how it overcomes the limitations of the state-of-the-art methods by optimizing task performance.

4.3 Preliminaries

Let $q \in \mathbb{R}^n$ be the configuration of a robot, where n is the number of degrees-of-freedom (DOFs). We introduce K task variables

$$\sigma_k = h_k(q) \in \mathbb{R}^{m_k}, \quad k \in \mathcal{K} \tag{79}$$

where m_k is the dimension of task k and $\mathcal{K} = \{1, 2, ..., K\}$ denotes a set of the indices. The hierarchy is defined such that σ_1 is at the top priority and σ_i is located higher in the priority order than σ_j if $i < j$. The task variables σ_k represent functional quantities (e.g., a cost or a potential function) as part of the desired actions, and h_k is a differentiable nonlinear function. Our goal is to devise a policy for the robotic system that would accomplish the K subtasks $\sigma_k, k \in \mathcal{K}$ in their descending order of importance. In the rest of the proposal, for notational compactness, the functional dependency on q is dropped whenever it is unambiguous.

Differentiating (79) with respect to time, we get

$$\dot{\sigma}_k = J_k(q)\dot{q}, \qquad J_k(q) = \frac{\partial h_k(q)}{\partial q} \tag{80}$$

where $J_k(q) \in \mathbb{R}^{m_k \times n}$ is the Jacobian matrix and $\dot{q}$ represents the velocity of the robot in the configuration space. This velocity can be computed by inverting the mapping (80) [92], [93], [94]. However, in the case of a redundant system i.e., when $n > m_k$, the problem (80) admits infinite solutions. A common approach is to solve for the minimum-norm velocity, which leads to the least-squares solution:

$$\dot{q} = J_k^\dagger(q)\dot{\sigma}_k$$

where $J_k^\dagger(q) = J_k^\top(q)(J_k(q)J_k^\top(q))^{-1}$ is the right pseudo-inverse of the Jacobian matrix $J_k(q)$. In the following $J_k(q)$ is assumed to be non-singular, hence of full row rank.

Similar to (80), the acceleration in the configuration space can be computed by further differentiating (80)

$$\ddot{\sigma}_k = J_k(q)\ddot{q} + \dot{J}_k(q)\dot{q}.$$

The minimum-norm solution for the acceleration $\ddot{q}$ is obtained as:

$$\ddot{q} = J_k^\dagger(q)\left(\ddot{\sigma}_k - \dot{J}_k(q)\dot{q}\right). \tag{81}$$

Equation (81) provides a basic method to compute system acceleration in an open-loop style. To improve convergence, a feedback term is added to (81), as suggested by [92] and [95], leading to the expanded form:

$$\ddot{q} = J_k^\dagger(q) \left(\left\{ \ddot{\sigma}_{k,d} + K_{p,k}\widetilde{\sigma}_k + K_{d,k}\frac{d\widetilde{\sigma}_k}{dt} \right\} - \dot{J}_k(q)\dot{q} \right) \tag{82}$$

where $\widetilde{\sigma}_k = \sigma_{k,d} - \sigma_k$ denotes the error between the desired task trajectory $\sigma_{k,d}$ and the actual task trajectory σ_k which can be computed from the system's current configurations using (79). For task k, the terms $K_{p,k}$ and $K_{d,k}$ are proportional and derivative gains, respectively, which shape the convergence of the error $\widetilde{\sigma}_k$. Equation (82) is called a *closed loop inverse kinematic* version of the equation (81). The controller having a similar structure is presented in [92].

Note that in the above control input computation technique, the desired task trajectory $\sigma_{k,d}$ is often chosen manually. Due to the complexity of the architecture, it is often difficult to select an appropriate desired trajectory $\sigma_{k,d}$ and low-level controller gains $K_{p,k}$, $K_{d,k}$ to optimize the overall system performance. Moreover, the above method is only a local optimization technique as opposed to global optimization techniques which minimize some performance index across a whole trajectory and typically offer better solutions compared to local optimization approaches.

4.4 Null-Space Projection

Consider a robotic system with a control-affine dynamics:

$$\dot{x}(t) = f(x) + G(x)u(t) \tag{83}$$

where $x(t)$ is the state of the system[3], $u(t)$ is the control input, $f(x)$ is a drift term, and $G(x)$ is the control coefficient. In the rest of the proposal, for notational compactness, the functional dependencies on x, and t are dropped whenever it is unambiguous. In the null-space projection technique, the control input u_2 corresponding to the second-priority task is projected onto the null space of the primary task using the formula:

$$u_2' = N_2(q)u_2 \tag{84}$$

where u_2' is the projected control input that does not interfere with the primary task. The null-space projector $N_2(q)$ is obtained by evaluating

$$N_2(q) = I - J_1^\dagger(q)J_1(q)$$

where $J_1^\dagger(q)$ is the right pseudo-inverse of the primary task's Jacobian J_1, and I is an identity matrix of suitable dimensions. Analogous to (84), for the remaining tasks in the hierarchy ($2 < k \leq K$), the control inputs are projected as $u_k' = N_k(q)u_k$, with the null-space projectors recursively computed as:

$$N_k(q) = N_{k-1}(q) \left(I - J_{k-1}^\dagger(q)J_{k-1}(q) \right) \quad 2 \leq k \leq K$$

and $N_1(q) = I$. Here I is the identity matrix with suitable dimensions. Each task input is computed as if it were acting alone; then before adding its contribution to the overall system control input, a lower-priority task is projected onto the null space of the immediately higher-priority task so as to remove those control input components that would conflict with it. This technique guarantees that lower-priority objectives are constrained and therefore do not interfere with higher-priority objectives. As a result, the high-priority task is always achieved, and the lower ones are met only if they do not conflict with the task of higher priority. The final control input can be formulated by adding up the primary task control input and all the projected control inputs:

$$u = u_1 + \sum_{k=2}^{K} u_k' = \sum_{k=1}^{K} N_k(q)u_k. \tag{85}$$

Plugging (85) into (83), we get

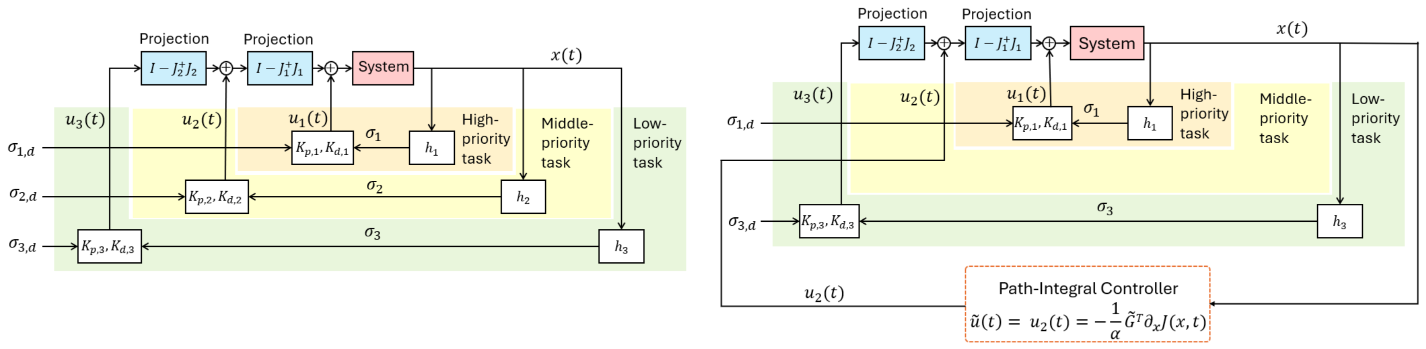
$$\dot{x}(t) = f(x) + G(x)\sum_{k\in\mathcal{K}} N_k(q)u_k. \tag{86}$$

A natural question arises of how many tasks can be handled simultaneously using this approach. Let us suppose that the primary task, of dimension m_1 is fulfilled by n DOFs robotic system. The null space of its Jacobian (of full row

[3]Often the case, the configuration of a system q is part of its states; see examples in Section 4.5.

rank) is a space of dimensions $n - m_1$. Supposing the secondary task of dimension m_2 does not conflict with the primary task (meaning that the secondary task acts in the null space of the primary task), the null space of their combination has dimension $n - m_1 - m_2$. Choosing the tasks in a way they are not conflicting, it is useful to add tasks until $\sum m_k = n$. Thus, once all the degrees of freedom of the system are covered, it is useless to add successive tasks of lower priority since they will be projected onto an empty space (thus giving always a null contribution to the system control input). In case of conflicting tasks, it is not possible to make any generic assumption regarding the useful number of tasks but, case by case, the intersection among null spaces should be analyzed.

4.4.1 Integration of Null Space Projection and Path Integral Control



(a) Conventional task hierarchical control

(b) Integration of null-space projection and path integral control (proposed architecture)

Figure 15: Comparison of the conventional task hierarchical control approach and our proposed control approach, which integrates null-space projection with path integral control.

In this section, we explain our proposed architecture. Consider a scenario where we need to accomplish K tasks, however, only the k-th task is handled using a path integral controller. Here, we assume only one task is accomplished using the path integral controller. A similar formulation can be extended to cases where multiple tasks are controlled using path integral controllers. The control inputs for the other tasks $u_i, i \neq k$ are designed using PD controllers (82). Using (86), the system dynamics can be written as

$$\dot{x}(t) = f(x) + \sum_{i \in \mathcal{K}, i \neq k} G(x) N_i u_i(t) + G(x) N_k u_k(t). \tag{87}$$

Defining $\widetilde{G}(x) = G(x)N_k$, $\widetilde{u} = u_k$ and

$$\widetilde{f}(x) = f(x) + \sum_{i \in \mathcal{K}, i \neq k} G(x) N_i u_i(t),$$

we can rewrite (87) as $dx = \widetilde{f}(x)dt + \widetilde{G}(x)\widetilde{u}\,dt$. Note that the terms $\widetilde{f}(x)$ and $\widetilde{G}(x)$ are computed using the PD-like low-level controllers (82) derived in Section 4.3 and the null-space projection explained in Section 4.4. Next, we derive a path integral controller for the control input $\widetilde{u}$.

First, we perturb $\widetilde{u}$ by adding a stochastic term: $\widetilde{u}\,dt \to \widetilde{u}\,dt + \hat{s}\,dw(t)$, where $w(t)$ is a standard Brownian motion and $\hat{s} \in \mathbb{R}$ is the diffusion coefficient. This perturbation allows the system to explore the policy space and discover a direction that minimizes the cost. The perturbed system dynamics are then given by

$$dx = \widetilde{f}(x)dt + \widetilde{G}(x)\left(\widetilde{u}\,dt + \hat{s}\,dw(t)\right). \tag{88}$$

Now we formulate the path integral control problem with the cost function:

$$\mathbb{E}_Q \left[\phi(x(T)) + \int_0^T \left(L(x(t)) + \frac{\alpha}{2} \|\widetilde{u}\|^2 \right) dt \right] \tag{89}$$

for some $\alpha > 0$. The expectation $\mathbb{E}_Q$ is taken with respect to the dynamics (88). The cost function has a quadratic control cost, an arbitrary state-dependent running cost $L(x(t))$, and a terminal cost $\phi(x(T))$. Consider the example of a platoon of robots illustrated in Section 4.1. Suppose task 2 (steering the platoon's centroid toward a goal position) is controlled by the path integral controller. Let there be I robots, with the configuration of each robot be denoted by $q_i, 1 \leq i \leq I$, and the goal position be q_g. In this case, the running cost $L(x(t))$ could be $\| \sum_{i=1}^{I} \frac{1}{I} q_i(t) - q_g \|^2$, and the terminal cost $\phi(x(T))$ could be $\| \sum_{i=1}^{I} \frac{1}{I} q_i(T) - q_g \|^2$.

We now formally state our problem:

Problem 4.

$$\min_{\widetilde{u}} \mathbb{E}_Q \left[\phi(x(T)) + \int_0^T \left(L(x(t)) + \frac{\alpha}{2} \|\widetilde{u}\|^2 \right) dt \right]$$
$$\text{s.t. } dx = \widetilde{f}(x)dt + \widetilde{G}(x) \left(\widetilde{u}\, dt + \hat{s}\, dw(t) \right).$$

It is well-known that the value function $J(x,t)$ of Problem 4 satisfies the Hamilton-Jacobi-Bellman (HJB) partial differential equation (PDE) [6]:

$$-\partial_t J = -\frac{1}{2\alpha} (\partial_x J)^\top \widetilde{G}\widetilde{G}^\top \partial_x J + L + \widetilde{f}^\top \partial_x J + \frac{\hat{s}^2}{2} \mathrm{Tr} \left(\widetilde{G}\widetilde{G}^\top \partial_x^2 J \right), \tag{90}$$

with the boundary condition $J(x(T),T) = \phi(x(T))$. The optimal control is expressed in terms of the solution to this PDE as follows:

$$\widetilde{u}(x,t) = -\frac{1}{\alpha} \widetilde{G}^\top \partial_x J\,(x,t)\,.$$

Therefore, in order to compute the optimal controls, we need to solve this backward-in-time PDE (90). Unfortunately, classical methods for solving partial differential equations (such as the finite difference method) of this nature suffer from the curse of dimensionality and are intractable for systems with more than a few state variables. The path integral control framework provides an alternative approach by transforming the HJB PDE into a path integral. This transformation allows us to approximate the solution using Monte Carlo simulations of the system's stochastic dynamics. We use the Feynman-Kac formula, which relates PDEs to path integrals [6]. First, we define a constant λ such that:

$$\hat{s}^2 = \frac{\lambda}{\alpha}. \tag{91}$$

Next, using this constant λ, we introduce the following transformed value function $\xi(x,t)$:

$$J(x,t) = -\lambda \log \left(\xi\,(x,t) \right). \tag{92}$$

The transformation (92) allows us to write the HJB PDE (90) in terms of $\xi(x,t)$ as

$$\partial_t \xi = \frac{L\xi}{\lambda} - \frac{\hat{s}^2}{2} \mathrm{Tr}\left(\widetilde{G}\widetilde{G}^\top \partial_x^2 \xi \right) + \frac{\hat{s}^2}{2\xi} (\partial_x \xi)^T \widetilde{G}\widetilde{G}^\top \partial_x \xi - \frac{\lambda}{2\alpha\xi} (\partial_x \xi)^\top \widetilde{G}\widetilde{G}^\top \partial_x \xi - \widetilde{f}^\top \partial_x \xi, \tag{93}$$

with the boundary condition $\xi(x(T),T) = \exp\left(-\frac{\phi(x(T))}{\lambda} \right)$. Using (91) in equation (93), we can rewrite the PDE (90) as a linear PDE in terms of $\xi(x,t)$ as:

$$\partial_t \xi = \frac{L\xi}{\lambda} - \widetilde{f}^\top \partial_x \xi - \frac{\hat{s}^2}{2} \mathrm{Tr} \left(\widetilde{G}\widetilde{G}^\top \partial_x^2 \xi \right) \tag{94}$$

with the boundary condition $\xi(x(T),T) = \exp\left(-\frac{\phi(x(T))}{\lambda} \right)$. This particular PDE is known as the *backward Chapman–Kolmogorov PDE*. Now we find the solution of the linearized PDE (94) using the Feynman-Kac lemma.

Theorem 10 (Feynman-Kac lemma)**.** *The solution to the linear PDE (94) exists. Moreover, the solution is unique in the sense that ξ solving (94) is given by*

$$\xi(x,t) = \mathbb{E}_P \left[\exp\left(-\frac{1}{\lambda} S(x,t) \right) \right] \tag{95}$$

where the expectation $\mathbb{E}_P$ is taken with respect to the uncontrolled dynamics of the system (88) (i.e., equation (88) with $\widetilde{u} = 0$) starting at x, t. $S(x,t)$ is the cost to go of the state-dependent cost of a trajectory given by

$$S(x,t) = \phi\,(x(T)) + \int_t^T L(x(t))dt.$$

Proof. The proof follows from [59, Theorem 9.1.1]. □

We now obtain the expression for the optimal control policy for Problem 4 via the following theorem.

Theorem 11. *The optimal solution of Problem 4 exists, is unique and is given by*

$$\widetilde{u}^*(x,t)dt = \mathcal{G}(x)\, \frac{\mathbb{E}_P\left[\exp\left(-\frac{1}{\lambda}S\right)\hat{s}\,\widetilde{G}(x)\,dw(t)\right]}{\mathbb{E}_P\left[\exp\left(-\frac{1}{\lambda}S\right)\right]}, \tag{96}$$

where the matrix $\mathcal{G}(x)$ *is defined as* $\mathcal{G}(x) = \widetilde{G}^\top(x)\left(\widetilde{G}(x)\widetilde{G}^\top(x)\right)^{-1}$.

Proof. The existence and uniqueness of the optimal solution follow from the existence and uniqueness of the linear PDE (94) (Theorem 10). The solution (96) can be computed by taking the gradient of (95) with respect to x [3, 6]. □

To evaluate expectations in (95) and (96) numerically, we discretize the uncontrolled dynamics and use Monte Carlo sampling [6]. After discretizing the uncontrolled dynamics, we get $x_{t+1} = x_t + \widetilde{f}(x_t)\Delta t + \hat{s}\,\widetilde{G}(x_t)\epsilon\sqrt{\Delta t}$, where the term ϵ is a time-varying vector of standard normal Gaussian random variables and Δt is the step size. The term $S(x,t)$ is approximately given as $S(x,t) \approx \phi(x_T) + \sum_{i=1}^{\tau} L(x_t)\Delta t$, where $\tau = \frac{T-t}{\Delta t}$. Now suppose we generate M samples of trajectories. Then we can approximate (96) as

$$\widetilde{u}^*(x,t) \approx \mathcal{G}(x_t)\, \frac{\sum_{j=1}^{M}\left[\exp\left(-\frac{1}{\lambda}S(x,t)\right)\hat{s}\,\widetilde{G}(x_t)\,\epsilon\right]}{\sum_{j=1}^{M}\left[\exp\left(-\frac{1}{\lambda}S(x,t)\right)\sqrt{\Delta t}\right]}. \tag{97}$$

Fig. 15 shows the comparison of the conventional task hierarchical control approach and our proposed control approach, which integrates null-space projection with path integral control.

Remark 4. *Note that as the number of samples* $M \to \infty$, (97) $\to$ (96) *i.e., path integral controller yields a globally optimal policy as* $M \to \infty$.

Remark 5. *Increasing the diffusion coefficient* $\hat{s}$ *allows the system to explore a wider range of possibilities, but it also introduces greater noise into the control input. Therefore,* $\hat{s}$ *should be carefully selected to strike an appropriate balance between exploration and control noise.*

The control input for each task in the hierarchy can be computed independently of the others. This allows for parallelization of the task hierarchy algorithm using GPUs, ensuring that the computational complexity remains manageable as additional tasks are introduced, provided a sufficient number of GPUs are available. As the number of agents increases, the dimensionality of the control inputs also grows. According to [7], in the case of the discrete-time path integral controller, the required sample size exhibits a *logarithmic* dependence on the dimension of the control input. However, a formal analysis of sample complexity for the continuous-time path integral controller remains an open area for future research.

4.5 Simulation Results

In this section, we present the simulation results for our proposed control framework.

4.5.1 Single Agent Example

Suppose a unicycle is navigating in a 2D space in the presence of an obstacle. The states of the unicycle model $x = [p_x\ p_y\ s\ \ \theta]^\top$ consist of its $x - y$ position $p := [p_x\ p_y]^\top$, speed s and heading angle $\theta \in [0, 2\pi]$. The configuration of the system $q := [p_x\ p_y\ \theta]^\top$ consists of its position p and the heading angle θ. The system dynamics are given by the following equation:

$$\begin{bmatrix}\dot{p}_x(t)\\ \dot{p}_y(t)\\ \dot{s}(t)\\ \dot{\theta}(t)\end{bmatrix} = \begin{bmatrix}s(t)\cos(\theta(t))\\ s(t)\sin(\theta(t))\\ 0\\ 0\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0\\ 1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}a(t)\\ \omega(t)\end{bmatrix}. \tag{98}$$

The control input $u := [a\ \omega]^\top$ consists of acceleration a and angular speed ω. The simulation is set with $T = 10$ seconds and $\Delta t = 0.01$ seconds. This unicycle system has to accomplish the following two tasks in descending order of importance: 1) obstacle avoidance and 2) move-to-goal. We design a proportional-derivative (PD) controller for the obstacle avoidance task and a path integral controller for the move-to-goal task.

Obstacle-avoidance

Suppose the obstacle is placed at $c = [c_x \;\; c_y]^\top$ having radius r. In the presence of an obstacle in the advancing direction, the robot aims to keep it at a safe distance from the obstacle. The obstacle avoidance task becomes active when the robot is within a certain threshold distance and moving toward the obstacle. The task variable for obstacle avoidance σ_1 is defined as:

$$\sigma_1 = \|p - c\|, \tag{99}$$

and the desired value of the task variable be $\sigma_{1,d} = r$. Differentiating (99) with respect to t, we get

$$\dot{\sigma}_1 = J_1 v \tag{100}$$

$$J_1 = \begin{bmatrix} \frac{p_x - c_x}{\|p-c\|} & \frac{p_y - c_y}{\|p-c\|} & 0 \end{bmatrix}, \quad v = \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ \omega \end{bmatrix}.$$

In order to devise the control input for task 1 $u_1 := [a_1 \; \omega_1]^\top$, we further differentiate (100) and get

$$\ddot{\sigma}_1 = \delta_1 + \Lambda_1 \begin{bmatrix} a_1 \\ \omega_1 \end{bmatrix}, \tag{101}$$

where δ_1 and Λ_1 are defined as:

$$\delta_1 = \frac{(p_y - c_y)^2 s^2 \cos^2\theta + (p_x - c_x)^2 s^2 \sin^2\theta}{\|p - c\|^3}, \tag{102}$$

$$\Lambda_1 = \begin{bmatrix} \frac{p_x - c_x}{\|p-c\|}\cos\theta + \frac{p_y - c_y}{\|p-c\|}\sin\theta \\ \frac{p_y - c_y}{\|p-c\|} s\cos\theta + \frac{p_x - c_x}{\|p-c\|} s\sin\theta \end{bmatrix}^\top. \tag{103}$$

Adding the feedback term to (101) similar to (82), we get

$$u_1 = \begin{bmatrix} a_1 \\ \omega_1 \end{bmatrix} = \Lambda_1^\dagger \left(\ddot{\sigma}_{1,d} + K_{p,1}\widetilde{\sigma}_1 + K_{d,1}\frac{d\widetilde{\sigma}_1}{dt} - \delta_1 \right)$$

where $\widetilde{\sigma}_1 = \sigma_{1,d} - \sigma_1$, and $K_{p,1}$ and $K_{d,1}$ are the proportional and derivative gains respectively for task 1.

Move-to-goal

In this task, the robot must reach the goal position $p_g := [p_{g,x} \; p_{g,y}]^\top$. The task variable σ_2 is given as

$$\sigma_2 = p = \begin{bmatrix} p_x \\ p_y \end{bmatrix} \tag{104}$$

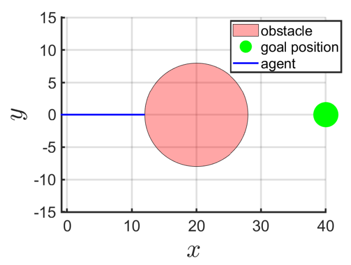
and the desired value of the task variable will be $\sigma_{2,d} = p_g$.

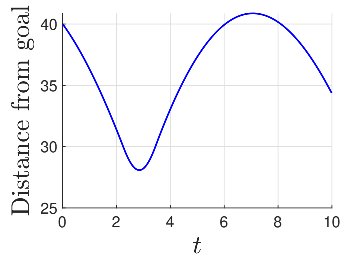
First, we design the control input $u_2 = [a_2 \; \omega_2]^\top$ using a PD controller similar to task 1. Differentiating (104) with respect to time, we get

$$\dot{\sigma}_2 = J_2 v, \quad J_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad v = \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ \omega \end{bmatrix}. \tag{105}$$

Differentiating (105) further we get

$$\ddot{\sigma}_2 = \Lambda_2 \begin{bmatrix} a_2 \\ \omega_2 \end{bmatrix}, \quad \Lambda_2 = \begin{bmatrix} \cos\theta & -s\sin\theta \\ \sin\theta & s\cos\theta \end{bmatrix}. \tag{106}$$





(a) Path followed without path integral controller (b) Distance from the goal over time

Figure 16: Results of single-agent example without the path integral controller

Adding the feedback term to (106) similar to (82), we get

$$u_2 = \begin{bmatrix} a_2 \\ \omega_2 \end{bmatrix} = \Lambda_2^{\dagger} \left(\ddot{\sigma}_{2,d} + K_{p,2}\widetilde{\sigma}_2 + K_{d,2}\frac{d\widetilde{\sigma}_2}{dt} \right)$$

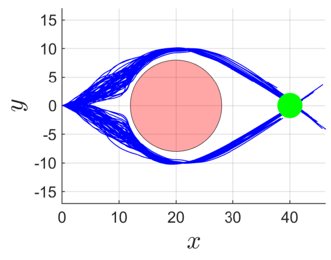
where $\widetilde{\sigma}_2 = \sigma_{2,d} - \sigma_2$, and $K_{p,2}$ and $K_{d,2}$ are the proportional and derivative gains respectively for task 2.

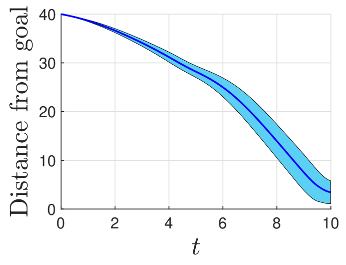
Next, we design the control input u_2 via the path integral controller. The perturbed dynamics are given by (88) with the diffusion coefficient $\hat{s} = 0.1$. We formulate the cost function (89) with

$$L(x(t)) = 0.07 \left\| p(t) - p_g \right\|, \ \phi(x(T)) = L(x(T)), \ \alpha = 10.$$

The number of Monte Carlo samples has been set to 10^4.

Lastly, the control input u_2 is projected onto the null space of task 1 and we get the final control input as $u = u_1 + (I - \Lambda_1^{\dagger}\Lambda_1)u_2$ where I is the identity matrix of suitable dimensions.





(a) Paths followed with path integral controller (b) Mean distance from goal $\pm$ standard deviation

Figure 17: Results of single-agent example using the path integral controller

Fig. 16(a) represents the results of the hierarchical controller when both tasks are controlled using PD controllers. As we can see, the robot initially moves towards the goal however, when it comes in the vicinity of the obstacle it drives itself away from the obstacle, as the obstacle avoidance task has a higher priority. The PD controllers get stuck in a local minima, the robot keeps oscillating as shown in Fig. 16(b) and never crosses the obstacle to reach the goal.

In Fig. 17, we show the results of the hierarchical controller when the first task is designed using a PD controller and the second task is designed using a path integral controller. Note that the path integral controller is designed by perturbing the control input $\widetilde{u}$ via a Brownian motion (88). Therefore, the outcome of the path integral controller is stochastic. In Fig. 17(a), we plot 100 trajectories obtained by the devised hierarchical controller. We can see that all the trajectories successfully avoid the obstacle by going around it and reaching the goal without getting stuck in local minimas. Fig. 17(b) shows the mean distance to the goal over time, along with the standard deviation across the 100 trajectories. The solid blue line represents the mean over the 100 trajectories and the sky-blue shaded area represents one standard deviation from the mean.

4.5.2 Two Agents Example

Suppose two unicycles are navigating in a 2D space in the presence of an obstacle. The unicycle dynamics follow the form in (98) where the state for each unicycle is defined as $x^{(i)} = [p_x^{(i)} \ \ p_y^{(i)} \ \ s^{(i)} \ \ \theta^{(i)}]^\top$, the configurations as

$q^{(i)} = [p_x^{(i)}\ p_y^{(i)}\ \theta^{(i)}]^\top$, and the control inputs as $u^{(i)} = [a^{(i)}\ \omega^{(i)}]^\top$ for $i = \{1, 2\}$. The simulation is set with $T = 10$ seconds and $\Delta t = 0.1$ seconds. This two-unicycle system has to accomplish the following three tasks in descending order of importance: 1) obstacle avoidance, 2) steering the centroid of the robots toward a goal position and 3) maintaining a specific distance between the two unicycles. We design a proportional-derivative (PD) controller for tasks 1) and 3), and a path integral controller for task 2).

Obstacle-avoidance

Suppose the obstacle is placed at $c = [c_x\ c_y]^\top$ having radius r. Similar to the single agent case, the obstacle avoidance task becomes active when any of the robots is within a certain threshold distance and moving toward the obstacle. Let $\sigma_1^{(i)}$ represent the task variable for the robot i and $\sigma_{1,d}$ represent the desired task value for both the robots. $\sigma_1^{(i)}$ and $\sigma_{1,d}$ can be written as

$$\sigma_1^{(i)} = \|p^{(i)} - c\|, \quad i = \{1, 2\}, \quad \sigma_{1,d} = r. \tag{107}$$

Similar to the single agent case, we differentiate (107) twice with respect to time for both robots and get

$$\ddot{\sigma}_1^{(i)} = \delta_1^{(i)} + \Lambda_1^{(i)} \begin{bmatrix} a_1^{(i)} \\ \omega_1^{(i)} \end{bmatrix} \tag{108}$$

where $\delta_1^{(i)}$ and $\Lambda_1^{(i)}$ can be defined similar to (102), (103):

Finally, adding the feedback term to (108) similar to (82), we get the control input for obstacle avoidance:

$$u_1^{(i)} = \begin{bmatrix} a_1^{(i)} \\ \omega_1^{(i)} \end{bmatrix} = \Lambda_1^{(i)\dagger}\left(\ddot{\sigma}_{1,d} + K_{p,1}^{(i)}\widetilde{\sigma}_1^{(i)} + K_{d,1}^{(i)}\frac{d\widetilde{\sigma}_1^{(i)}}{dt} - \delta_1^{(i)}\right)$$

where $\widetilde{\sigma}_1^{(i)} = \sigma_{1,d} - \sigma_1^{(i)}$, and $K_{p,1}^{(i)}$ and $K_{d,1}^{(i)}$ are the proportional and derivative gains respectively for task 1 and for $i = \{1, 2\}$. Also, note that

$$\Lambda_1 = \begin{bmatrix} \Lambda_1^{(1)} & 0 \\ 0 & \Lambda_1^{(2)} \end{bmatrix}, \quad \Lambda_1^\dagger = \begin{bmatrix} \Lambda_1^{(1)\dagger} & 0 \\ 0 & \Lambda_1^{(2)\dagger} \end{bmatrix}.$$

Steering the robots' centroid toward a goal position

This task involves steering the centroid of the robots toward a goal position p_g. The task variable σ_2 is given as

$$\sigma_2 = \frac{p^{(1)} + p^{(2)}}{2} \tag{109}$$

and the desired value of the task variable will be $\sigma_{2,d} = p_g$.

First, we design the control input $u_2 = [a_2^{(1)}\ \omega_2^{(1)}\ a_2^{(2)}\ \omega_2^{(2)}]^\top$ using a PD controller. Differentiating (109) twice with respect to time we get

$$\ddot{\sigma}_2 = \Lambda_2 \begin{bmatrix} a_2^{(1)} \\ \omega_2^{(1)} \\ a_2^{(2)} \\ \omega_2^{(2)} \end{bmatrix}, \ \Lambda_2 = \frac{1}{2}\begin{bmatrix} \cos\theta^{(1)} & \sin\theta^{(1)} \\ -s^{(1)}\sin\theta^{(1)} & s^{(1)}\cos\theta^{(1)} \\ \cos\theta^{(2)} & \sin\theta^{(2)} \\ -s^{(2)}\sin\theta^{(2)} & s^{(2)}\cos\theta^{(2)} \end{bmatrix}^\top. \tag{110}$$

Adding the feedback term to (110) similar to (82), we get

$$u_2 = \begin{bmatrix} a_2^{(1)} \\ \omega_2^{(1)} \\ a_2^{(2)} \\ \omega_2^{(2)} \end{bmatrix} = \Lambda_2^\dagger\left(\ddot{\sigma}_{2,d} + K_{p,2}\widetilde{\sigma}_2 + K_{d,2}\frac{d\widetilde{\sigma}_2}{dt}\right)$$

where $\widetilde{\sigma}_2 = \sigma_{2,d} - \sigma_2$, and $K_{p,2}$ and $K_{d,2}$ are the proportional and derivative gains respectively for task 2.

Next, we design the control input for task 2) via a path integral controller. The perturbed dynamics are given by (88) with the diffusion coefficient $\hat{s} = 0.1$. We formulate the cost function (89) with $\alpha = 10$,

$$L(x(t)) = 0.21 \left\| \frac{p^{(1)}(t) + p^{(2)}(t)}{2} - p_g \right\|, \; \phi(x(T)) = L(x(T)).$$

The number of Monte Carlo samples has been set to 10^4.

Maintaining a specific distance between two unicycles

In this task, the two unicycles must maintain the distance l between each other. The task variable σ_3 is given as

$$\sigma_3 = \frac{1}{2}\left(p^{(1)} - p^{(2)}\right)^\top \left(p^{(1)} - p^{(2)}\right) \tag{111}$$

and the desired value of the task variable will be $\sigma_{3,d} = \frac{l^2}{2}$. Differentiating (111) twice with respect to time we get

$$\ddot{\sigma}_3 = \delta_3 + \Lambda_3 \begin{bmatrix} a_3^{(1)} \\ \omega_3^{(1)} \\ a_3^{(2)} \\ \omega_3^{(2)} \end{bmatrix} \tag{112}$$

$$\delta_3 = s^{(1)^2} - 2s^{(1)}s^{(2)}\left(\cos\theta^{(1)}\cos\theta^{(2)} + \sin\theta^{(1)}\sin\theta^{(2)}\right) + s^{(2)^2},$$

$$\Lambda_3 = \begin{bmatrix} \left(p_x^{(1)} - p_x^{(2)}\right)\cos\theta^{(1)} + \left(p_y^{(1)} - p_y^{(2)}\right)\sin\theta^{(1)} \\ \left(p_x^{(2)} - p_x^{(1)}\right)s^{(1)}\sin\theta^{(1)} + \left(p_y^{(1)} - p_y^{(2)}\right)s^{(1)}\cos\theta^{(1)} \\ \left(p_x^{(2)} - p_x^{(1)}\right)\cos\theta^{(2)} + \left(p_y^{(2)} - p_y^{(1)}\right)\sin\theta^{(2)} \\ \left(p_x^{(1)} - p_x^{(2)}\right)s^{(2)}\sin\theta^{(2)} + \left(p_y^{(2)} - p_y^{(1)}\right)s^{(2)}\cos\theta^{(2)} \end{bmatrix}^\top.$$

Finally, adding the feedback term to (112) similar to (82), we get

$$u_3 = \begin{bmatrix} a_3^{(1)} \\ \omega_3^{(1)} \\ a_3^{(2)} \\ \omega_3^{(2)} \end{bmatrix} = \Lambda_3^\dagger \left(\ddot{\sigma}_{3,d} + K_{p,3}\widetilde{\sigma}_3 + K_{d,3}\frac{d\widetilde{\sigma}_3}{dt} - \delta_3\right)$$

where $\widetilde{\sigma}_3 = \sigma_{3,d} - \sigma_3$, and $K_{p,3}$ and $K_{d,3}$ are the proportional and derivative gains respectively for task 3.

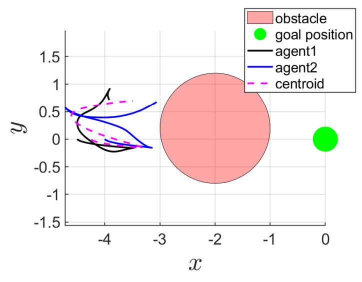
Lastly, the control input of the second task u_2 is projected onto the null space of task 1, and the control input of task 3 u_3 is projected onto the null spaces of both task 1 and task 2. This yields the overall control input as:

$$u = u_1 + (I - \Lambda_1^\dagger \Lambda_1)\left(u_2 + (I - \Lambda_2^\dagger \Lambda_2)u_3\right)$$

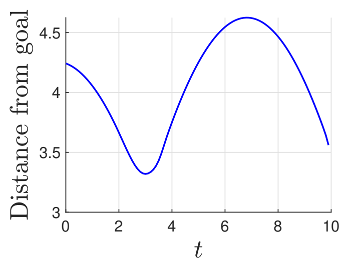
where I is the identity matrix of suitable dimensions.

In this experiment, we set the desired distance between the agents to $l = 0.5$ with the initial positions of the unicycles at $[p_x^{(1)} \; p_y^{(1)} \; p_x^{(2)} \; p_y^{(2)}]^\top = [-4.5 \;\; 0 \;\; -4 \;\; 0]^\top$. Fig. 18(a) represents the results of the hierarchical controller when all three tasks are controlled using PD controllers. As we can see, the centroid of the two unicycles initially moves towards the goal however, when the unicycles come in the vicinity of the obstacle they drive themselves away from the obstacle, as the obstacle avoidance task has a higher priority. Due to the limitations of the PD controllers, which get stuck in local minima, the unicycles begin to oscillate and never successfully cross the obstacle to reach the goal, as seen in Fig. 18(b).

In Fig. 19, we present the results using the hierarchical controller, where tasks 1) and 3) are managed with PD controllers, while task 2) is handled using a path integral controller. In Fig. 19(a), we plot 100 trajectories of the two agents and their centroid. The hierarchical controller successfully ensures obstacle avoidance and steers the centroid towards the goal in all trajectories, without getting stuck in local minima. Fig. 19(b) shows the mean distance of the centroid from the goal over time, along with the standard deviation across the 100 trajectories. The solid blue line represents the mean and the sky-blue shaded area represents one standard deviation from the mean.

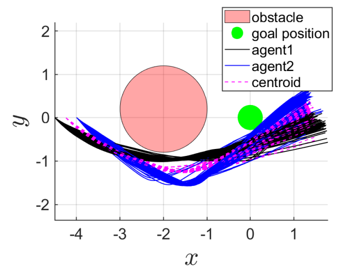


(a) Path followed without path integral controller

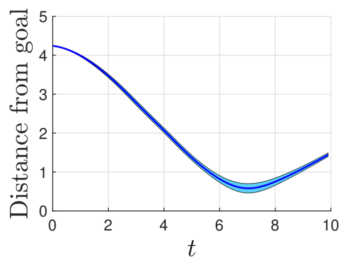


(b) Distance from the goal over time

Figure 18: Results of two-agents example without the path integral controller



(a) Paths followed with path integral controller



(b) Mean distance from goal ± standard deviation

Figure 19: Results of two-agents example using the path integral controller

In Fig. 20(a), we show the distance between the agents over time when the controller for task 2) is designed using a PD controller. Fig. 20(b) shows the mean distance between the agents over time, using the path integral controller for task 2). The solid blue line represents the mean across 100 trajectories, while the shaded area corresponds to one standard deviation. This figure highlights that the path integral controller helps the agents better maintain the desired distance between each other compared to the PD controller.

4.6 Publications

- **A. Patil**, R. Funada, T. Tanaka, L. Sentis, "Task Hierarchical Control via Null-Space Projection and Path Integral Approach ," *submitted to 2023 American Control Conference (ACC)*
- M. Baglioni, **A. Patil**, L. Sentis, A. Jamshidnejad "Achieving Multi-UAV Best Viewpoint Coordination in Obstructed Environments," *submitted to IEEE Control Systems Letters (L-CSS)*
- M. Baglioni, **A. Patil**, L. Sentis, A. Jamshidnejad "Achieving Multi-UAV Best Viewpoint Coordination in Obstructed Environments," *submitted to 2023 American Control Conference (ACC)*

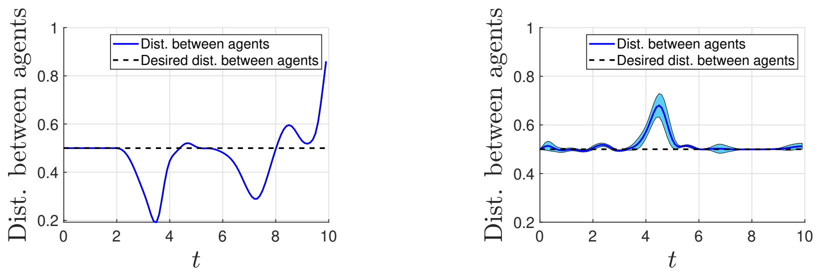
4.7 Future Work

For future work, we aim to incorporate the importance sampling techniques into the path integral controller, which offers improved sample efficiency compared to the standard approach. Additionally, we plan to extend the application of this framework to large-scale multi-robot systems and robotic manipulators with a high number of degrees of freedom.

5 Deceptive Control

5.1 Motivation

We consider a deception problem between a supervisor and an agent. The supervisor delegates an agent to perform a certain task and provides a reference policy to be followed in a stochastic environment. The agent, on the other hand, aims to achieve a different task and may deviate from the reference policy to accomplish its own task. The agent uses a



(a) Without path integral controller: Distance between agents

(b) With path integral controller: Mean distance between agents $\pm$ standard deviation

Figure 20: Distance between agents over time

deceptive policy to hide its deviations from the reference policy. In this work, we synthesize the optimal policies for such a deceptive agent.

We formulate the agent's deception problem using motivations from hypothesis testing theory. We assume that the supervisor aims to detect whether the agent deviated from the reference policy by observing the state-action paths of the agent. On the flip side, the agent's goal is to employ a deceptive policy that achieves the agent's task and minimizes the detection rate of the supervisor. We design the agent's deceptive policy that minimizes the Kullback-Leibler (KL) divergence from the reference policy while achieving the agent's task. The use of KL divergence is motivated by the log-likelihood ratio test, which is the most powerful detection test for any given significance level [96]. Minimizing the KL divergence is equivalent to minimizing the expected log-likelihood ratio between distributions of the paths generated by the agent's deceptive policy and the reference policy. We also note that due to the Bratagnolle-Huber inequality [97], for any statistical test employed by the supervisor, the sum of false positive and negative rates is lower bounded by a decreasing function of KL divergence between the agent's policy and the reference policy. Consequently, minimizing the KL divergence is a proxy for minimizing the detection rate of the supervisor. We represent the agent's task with a cost function and formulate the agent's objective function as a weighted sum of the cost function and the KL divergence.

We assume that the agent's environment follows discrete-time continuous-state dynamics. When the dynamics are linear, the supervisor's (stochastic) policies are Gaussian, and the cost functions are quadratic, minimizing a weighted sum of the cost function, and the KL divergence leads to solving a linear quadratic regulator problem. However, we consider a broader setting with potentially non-linear state dynamics, non-quadratic cost functions, and non-Gaussian reference policies. In this case, the agent's optimal deceptive policy does not necessarily admit a closed-form solution. While the agent's problem can be solved using backward dynamic programming, this approach suffers from the curse of dimensionality.

We show that, under the assumption of deterministic state dynamics, the optimal deceptive actions can be generated using path integral control without explicitly synthesizing a policy. In detail, we propose a two-step randomized algorithm for simulator-driven control for deception. At each time step, the algorithm first creates forward Monte Carlo samples of system paths under the reference policy. Then, the algorithm uses a cost-proportional weighted sampling method to generate a control input at that time step. We show that the proposed approach asymptotically converges to the optimal action distribution. Since Monte Carlo simulations can be efficiently parallelized, our approach allows the agent to generate the optimal deceptive actions online.

5.2 Literature Review

Deception naturally occurs in settings where two parties with conflicting objectives coexist. The example domains for deception include robotics [98, 99], supervisory control settings [100, 101], warfare [102], and cyber systems [103].

We formulate a deception problem motivated by hypothesis testing. This problem has been studied for fully observable Markov decision processes [100], partially observable Markov decision processes [101], and hidden Markov models [104]. Different from [100, 101, 104] that study discrete-state systems and directly solve an optimization problem for the synthesis of deceptive policies, we consider a nonlinear continuous-state system and provide a sampling-based solution for the synthesis of deceptive policies. In the security framework, [105, 106] study the detectability of an attacker in a stochastic control setting. Similar to our formulation, [105, 106] provide a KL divergence-based optimization problem. While we consider an agent whose goal is to optimize a different cost function from the supervisor, [105, 106] consider an attacker whose goal is to maximize the state estimation error of a controller.

KL divergence objective is also used in reinforcement learning [107, 108] to improve the learning performance and in KL control frameworks [109, 110] for the efficient computation of optimal policies. In [110], Ito et al. studied the KL control problem for nonlinear continuous-state space systems and characterized the optimal policies. Different from [110], we provide a randomized control algorithm based on path integral approach that converges to the optimal policy as the number of samples increases. Path integral control is a sampling-based algorithm employed to solve nonlinear stochastic optimal control problems numerically [2, 3, 4]. It allows the policy designer to compute the optimal control inputs online using Monte Carlo samples of system paths. The Monte Carlo simulations can be massively parallelized on GPUs, and thus the path integral approach is less susceptible to the curse of dimensionality [6].

5.3 Contributions

The contributions of this work are as follows:

1. The work studies a problem of deception under supervisory control for continuous-state discrete-time stochastic systems. Given a reference policy, we formalize the synthesis of an optimal deceptive policy as a KL control problem and solve it using backward dynamic programming.

2. For the deterministic state dynamics, we propose a path-integral-based solution methodology for simulator-driven control. We develop an algorithm based on Monte Carlo sampling to numerically compute the optimal deceptive actions online. Furthermore, we show that the proposed approach asymptotically converges to the optimal control distribution of the deceptive agent.

3. We present a numerical example to validate the derived simulator-driven control synthesis framework.

Notations

Let $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ be a measurable space where $\mathcal{X} \subseteq \mathbb{R}^n$ is a Borel set and $\mathcal{B}(\mathcal{X})$ is a Borel σ-algebra. Suppose $(\Omega, \mathcal{F}, \mathcal{P})$ is a probability space. An $(\mathcal{F}, \mathcal{B}(\mathcal{X}))$-measurable random variable X is a function $X : \Omega \to \mathcal{X}$ whose probability distribution P_X is defined by

$$P_X(B) = \mathcal{P}(X \in B) \quad \forall B \in \mathcal{B}(\mathcal{X})$$

$P_{X_2|X_1}(\cdot|\cdot) : \mathcal{B}(\mathcal{X}_2) \times \mathcal{X}_1 \to [0,1]$ represents a stochastic kernel on $\mathcal{X}_2$ given $\mathcal{X}_1$. For simplicity, we write $P_X(dx)$ and $P_{X_2|X_1}(dx_2|x_1)$ as $P(dx)$ and $P(dx_2|x_1)$. If P_1 and P_2 are probability distributions on $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ then, the Kullback-Leibler (KL) divergence from P_1 to P_2 is defined as

$$D(P_2 \| P_1) = \int_{\mathcal{X}} \log \frac{dP_2}{dP_1}(x) P_2(dx)$$

if the Radon-Nikodym derivative $\frac{dP_2}{dP_1}$ exists, and $D(P_2\|P_1) = +\infty$ otherwise. Throughout this proposal, we use the natural logarithm. Let $\mathcal{T} = \{0, 1, ..., T\}$ be the set of discrete time indices. A set of variables $\{x_0, x_1, \dots, x_T\}$ is denoted by $x_{0:T}$ and a Cartesian product of sets $\mathcal{X}_0 \times \mathcal{X}_1 \times \dots \times \mathcal{X}_T$ is denoted by $\mathcal{X}_{0:T}$. $P_{X_{0:T}}(dx_{0:T})$ denotes the joint probability distribution of random variables $X_0, X_1, \dots, X_T$ on $(\mathcal{X}_{0:T}, \mathcal{B}(\mathcal{X}_{0:T}))$.

5.4 Problem Formulation

We consider a setting in which a supervisor contracts an agent to perform a certain task. Suppose the agent operates in a stochastic environment and follows discrete-time continuous-state dynamics. Let the state transition law of the agent be denoted by $P(dx_{t+1}|x_t, u_t) : \mathcal{B}(\mathcal{X}_{t+1}) \times \mathcal{X}_t \times \mathcal{U}_t \to [0,1]$, where $\mathcal{X}_t \subseteq \mathbb{R}^n$ and $\mathcal{U}_t \subseteq \mathbb{R}^m$ be the spaces of states and control inputs at time step $t \in \mathcal{T} := \{0, 1, \cdots, T\}$, respectively. Suppose a supervisor provides a (possibly stochastic) reference policy $\{R_{U_t|X_t}(\cdot|x_t)\}_{t=0}^{T-1}$ to the agent and expects the agent to follow the policy to accomplish a certain task. Here, $R_{U_t|X_t} : \mathcal{B}(\mathcal{U}_t) \times \mathcal{X}_t \to [0,1]$ is a stochastic kernel on $\mathcal{U}_t$ given $\mathcal{X}_t$. The agent, on the other hand, aims to achieve a different task by minimizing the following cost function, which we henceforth call as *path cost*:

$$C_{0:T}(x_{0:T}, u_{0:T-1}) := \sum_{t=0}^{T-1} C_t(x_t, u_t) + C_T(x_T) \tag{113}$$

where $C_t(\cdot,\cdot) : \mathcal{X}_t \times \mathcal{U}_t \to \mathbb{R}$ for $t \in \mathcal{T}$ and $C_T(\cdot) : \mathcal{X}_T \to \mathbb{R}$ represent the stage costs and the terminal cost, respectively. In order to minimize the path cost (113), the agent designs its policy (possibly stochastic) $\{Q_{U_t|X_t}(\cdot|x_t)\}_{t=0}^{T-1}$

that may deviate from the reference policy $\{R_{U_t|X_t}(\cdot|x_t)\}_{t=0}^{T-1}$. The agent also attempts to be stealthy to hide its deviations from the supervisor. While the agent executes its policy Q, suppose the supervisor observes its state-action paths $\{x_{0:T}, u_{0:T-1}\}$, and uses a likelihood ratio test to detect whether the agent deviates from the reference policy. According to the Neyman–Pearson lemma, the likelihood-ratio test is optimal among all simple hypothesis tests for a given significance level [96]. In other words, we consider the worst-case scenario for the agent to be detected by the supervisor. Suppose the initial state $X_0 = x_0$ of the agent is known. We denote the joint probability distribution of the state-action paths induced via the reference policy by

$$R_{X_{0:T}\times U_{0:T-1}}(dx_{0:T}\times du_{0:T-1}) = \prod_{t=0}^{T-1} P(dx_{t+1}|x_t,u_t)R(du_t|x_t), \tag{114}$$

and the joint distribution induced via the agent's policy by

$$Q_{X_{0:T}\times U_{0:T-1}}(dx_{0:T}\times du_{0:T-1}) = \prod_{t=0}^{T-1} P(dx_{t+1}|x_t,u_t)Q(du_t|x_t). \tag{115}$$

Given a path $\{x_{0:T}, u_{0:T-1}\}$ that is randomly sampled under the agent's policy, the supervisor computes the log-likelihood ratio (LLR)

$$\pi(x_{0:T}, u_{0:T-1}) = \log \frac{dQ_{X_{0:T}\times U_{0:T-1}}}{dR_{X_{0:T}\times U_{0:T-1}}}(x_{0:T}, u_{0:T-1}). \tag{116}$$

The supervisor decides that the agent uses the reference policy R if $\pi(x_{0:T}, u_{0:T-1}) \leq c$, and deviates from R otherwise. Here c is a constant chosen by the supervisor to obtain a specified significance level. An agent not wanting to be detected by the supervisor must minimize the LLR (116). However, since the agent's trajectories are stochastic, the agent cannot directly minimize the LLR. We consequently consider that the agent's goal is to minimize the expected LLR as follows:

$$\Pi = \mathbb{E}_Q\left[\log \frac{dQ_{X_{0:T}\times U_{0:T-1}}}{dR_{X_{0:T}\times U_{0:T-1}}}(x_{0:T}, u_{0:T-1})\right] \tag{117}$$

where $\mathbb{E}_Q[\cdot]$ represents the expectation with respect to the probability distribution Q (115). Note that equation (117) also defines the Kullback-Leibler (KL) divergence $D(Q\|R)$ between the agent's distribution Q and the reference distribution R. Now we show that $D(Q\|R)$ can be written as the stage-additive KL divergence between $Q_{U_t|X_t}$ and $R_{U_t|X_t}$:

$$\begin{aligned}
&\int_B Q_{X_{0:T}\times U_{0:T-1}}(dx_{0:T}\times du_{0:T-1}) \\
&= \int_B \prod_{t=0}^{T-1} P(dx_{t+1}|x_t,u_t)Q(du_t|x_t) && \text{(118a)}\\
&= \int_B \left(\prod_{t=0}^{T-1} \frac{dQ_{U_t|X_t}}{dR_{U_t|X_t}}(x_t,u_t)\right)\prod_{t=0}^{T-1} P(dx_{t+1}|x_t,u_t)R(du_t|x_t) && \text{(118b)}\\
&= \int_B \left(\prod_{t=0}^{T-1} \frac{dQ_{U_t|X_t}}{dR_{U_t|X_t}}(x_t,u_t)\right)R_{X_{0:T}\times U_{0:T-1}}(dx_{0:T}\times du_{0:T-1}) && \text{(118c)}
\end{aligned}$$

where, B is a Borel set belonging to the $\sigma-$algebra $\mathcal{B}(\mathcal{X}_{0:T}\times\mathcal{U}_{0:T-1})$. The first equality (118a) follows from the definition (115), the second equality (118b) by the definition of the Radon-Nikodym derivative [1] and the last one (118c) from the definition (114). Using (118), we can write the Radon-Nikodym derivative $\frac{dQ_{X_{0:T}\times U_{0:T-1}}}{dR_{X_{0:T}\times U_{0:T-1}}}$ as follows:

$$\frac{dQ_{X_{0:T}\times U_{0:T-1}}}{dR_{X_{0:T}\times U_{0:T-1}}}(x_{0:T}, u_{0:T-1}) = \prod_{t=0}^{T-1} \frac{dQ_{U_t|X_t}}{dR_{U_t|X_t}}(x_t,u_t)\,. \tag{119}$$

Using (119), we get the following:

$$
\begin{aligned}
D(Q\|P) &= \mathbb{E}_Q\left[\log \frac{dQ_{X_{0:T}\times U_{0:T-1}}}{dR_{X_{0:T}\times U_{0:T-1}}}(x_{0:T}, u_{0:T-1})\right] \\
&= \mathbb{E}_Q\left[\log \prod_{t=0}^{T-1} \frac{dQ_{U_t|X_t}}{dR_{U_t|X_t}}(x_t, u_t)\right] \\
&= \mathbb{E}_Q\left[\sum_{t=0}^{T-1} \log \frac{dQ_{U_t|X_t}}{dR_{U_t|X_t}}(x_t, u_t)\right] && (120) \\
&= \mathbb{E}_Q\left[\sum_{t=0}^{T-1} D(Q_{U_t|X_t}(\cdot|X_t)\|R_{U_t|X_t}(\cdot|X_t))\right]. && (121)
\end{aligned}
$$

Since the KL divergence $D(Q\|R)$ is equivalent to the expected LLR (117), in this work, the KL divergence is used as a proxy for the measure of the agent's deviations from the reference policy.

Minimizing the KL divergence is in fact equivalent to minimizing the detection rate of an attacker for an ergodic process as proved in [106]. While we do not consider an ergodic process, the use of KL divergence is still well-motivated by the Bretagnolle–Huber inequality [97]. Let $\mathcal{E}$ be an arbitrary set of events that the supervisor will identify the agent as a deceptive agent, i.e., the agent followed Q. According to the Bretagnolle–Huber inequality, we have

$$\Pr(\mathcal{E}|R) + \Pr(\neg\mathcal{E}|Q) \geq \frac{1}{2}\exp(-D(Q||R)) \tag{122}$$

where $\Pr(\mathcal{E}|R)$ and $\Pr(\neg\mathcal{E}|Q)$ denote the supervisor's false positive and negative rates, respectively. The false positive rate is the probability that the supervisor will identify the well-intentioned agent as a deceptive agent, i.e., the agent's policy is R, but the supervisor thinks that the agent has followed Q. Similarly, the false negative rate is the probability that the supervisor will identify the deceptive agent as a well-intentioned agent. The Bretagnolle–Huber inequality (122) states that the sum of the supervisor's false positive and negative rates is lower bounded by a decreasing function of the KL divergence between the distributions Q and R. Therefore, an agent wanting to increase the supervisor's false classification rate should minimize the KL divergence from R to Q.

The goal of the agent is to design a deceptive policy Q that minimizes the expected path cost $\mathbb{E}_Q\left[C_{0:T}(X_{0:T}, U_{0:T-1})\right]$ (113) while limiting the KL divergence $D\left(Q\|R\right)$ (121). Using (113) and (121), we propose the following KL control problem for the synthesis of optimal deceptive policies for the agent:

Problem 5 (Synthesis of optimal deceptive policy)**.**

$$\min_{\{Q_{U_t|X_t}\}_{t=0}^{T-1}} \mathbb{E}_Q \sum_{t=0}^{T-1} \left\{C_t(X_t, U_t) + \lambda D(Q_{U_t|X_t}(\cdot|X_t)\|R_{U_t|X_t}(\cdot|X_t))\right\} + \mathbb{E}_Q C_T(X_T) \tag{123}$$

where λ is a positive weighting factor that balances the trade-off between the KL divergence and the path cost.

We explain the above KL control problem via the following example:

Example 4. *Consider a drone that is contracted by a supervisor to perform a surveillance task over an area. The supervisor prefers the drone to operate at high speeds (policy R) to improve the efficiency of the surveillance. The operator of the drone, the agent, on the other hand, prefers the drone to operate in a battery-saving, safe mode (policy Q) to improve the longevity of the drone. The agent does not want to get detected by the supervisor and fired. Hence, the goal of the agent is to operate in a way that would balance the energy consumption ($\mathbb{E}_Q\left[C_{0:T}(X_{0:T}, U_{0:T-1})\right]$) and the deviations from the behavior desired by the supervisor ($D(Q||R)$).*

5.5 Synthesis of Optimal Control Policies

In this section, we solve Problem 5 using backward dynamic programming and propose a policy synthesis algorithm based on path integral control.

5.5.1 Backward Dynamic Programming

Notice that the cost function of Problem 5 possesses the time-additive Bellman structure and, therefore, can be solved by utilizing the principle of dynamic programming [111]. Define for each $t \in \mathcal{T}$ and $x_t \in \mathcal{X}_t$, the value function:

$$J_t(x_t) := \inf_{\{Q_{U_k|X_k}\}_{k=t}^{T-1}} \mathbb{E}_Q \sum_{k=t}^{T-1} \Big\{ C_k(X_k, U_k) + \lambda D(Q_{U_k|X_k}(\cdot|X_k) \| R_{U_k|X_k}(\cdot|X_k)) \Big\} + \mathbb{E}_Q C_T(X_T).$$

Notice that in (124), we used "inf" instead of "min" since we do not know if the infimum is attained. In the following theorem, we show that the infimum is indeed attained, and therefore, "inf" can be replaced by "min".

Theorem 12. *The value function $J_t(x_t)$ satisfies the following backward Bellman recursion with the terminal condition $J_T(x_T) = C_T(x_T)$:*

$$J_t(x_t) = -\lambda \log \left\{ \int_{\mathcal{U}_t} \exp\left(-\frac{C_t(x_t, u_t)}{\lambda} \right) \times \exp\left(-\frac{1}{\lambda} \int_{\mathcal{X}_{t+1}} J_{t+1}(x_{t+1}) P(dx_{t+1}|x_t, u_t) \right) R(du_t|x_t) \right\}$$

and the minimizer of Problem 5 is given by

$$Q^*_{U_t|X_t}(B_{U_t}|x_t) = \frac{\int_{B_{U_t}} \exp(-\rho_t(x_t, u_t)/\lambda) R(du_t|x_t)}{\int_{\mathcal{U}_t} \exp(-\rho_t(x_t, u_t)/\lambda) R(du_t|x_t)} \tag{124}$$

where

$$\rho_t(x_t, u_t) := C_t(x_t, u_t) + \int_{\mathcal{X}_{t+1}} J_{t+1}(x_{t+1}) P(dx_{t+1}|x_t, u_t) \tag{125}$$

and B_{U_t} is a Borel set belonging to the $\sigma-$algebra $\mathcal{B}(\mathcal{U}_t)$.

Proof. By Bellman's optimality principle, the value function satisfies the following recursive relationship:

$$J_t(x_t) = \inf_{Q_{U_t|X_t}} \int_{\mathcal{U}_t} \Big\{ \rho_t(x_t, u_t) + \lambda \log \frac{dQ}{dR}(u_t|x_t) \Big\} Q(du_t|x_t) \tag{126}$$

where $\rho_t(x_t, u_t)$ is defined by (125). Invoking the Legendre duality between the KL divergence and free energy (see Appendix 11.2), it can be shown that there exists a minimizer $Q^*_{U_t|X_t}$ of the right-hand side of (126), which can be written as

$$Q^*_{U_t|X_t}(B_{U_t}|x_t) = \frac{\int_{B_{U_t}} \exp(-\rho_t(x_t, u_t)/\lambda) R(du_t|x_t)}{\int_{\mathcal{U}_t} \exp(-\rho_t(x_t, u_t)/\lambda) R(du_t|x_t)} \tag{127}$$

where B_{U_t} is a Borel set belonging to the $\sigma-$algebra $\mathcal{B}(\mathcal{U}_t)$. Using (127), the value of (126) can be computed as

$$J_t(x_t) = -\lambda \log \left\{ \int_{\mathcal{U}_t} \exp\left(-\frac{\rho_t(x_t, u_t)}{\lambda} \right) R(du_t|x_t) \right\}. \tag{128}$$

Substituting (125) into (128), we obtain the recursive expression (124). □

Theorem 12 provides a recursive method to compute the value functions $J_t(x_t)$ and optimal control distributions $Q^*_{U_t|X_t}$ backward in time. As one can see, to perform the backward recursions (124) and (124), the function $J_t(x_t)$ must be evaluated everywhere in the continuous domain $\mathcal{X}_t$. Therefore, in practice, an exact implementation of backward dynamic programming is computationally costly (unless the problem has a special structure, for example, linear state dynamics and quadratic costs). The computational cost grows quickly with the dimension of the state space of the system, which is referred to as the curse of dimensionality. In the next section (Section 5.5.2), we show that under the assumption of the deterministic state transition law, the backward Bellman recursions can be linearized. This allows us to design a simulator-driven algorithm to compute optimal deceptive actions.

5.5.2 Path Integral Solution

In this section, we focus on a special case in which the agent's dynamics are deterministic and propose a simulator-driven algorithm to compute the optimal deceptive actions via path integral control.

Assumption 5. *The state transition law is governed by a deterministic mapping $F_t : \mathcal{X}_t \times \mathcal{U}_t \to \mathcal{X}_{t+1}$ as*

$$x_{t+1} = F_t(x_t, u_t); \tag{129}$$

that is, $P(dx_{t+1}|x_t, u_t) = \delta_{F_t(x_t,u_t)}(dx_{t+1})$, where δ denotes the Dirac measure.

Remark 6. *Note that under Assumption 5, the agent can deviate from the reference policy R only if it is stochastic. Otherwise, under any deviations from the reference policy, with a positive probability, the supervisor will be sure that the agent did not follow the reference policy. Therefore, in what follows, we consider the reference policy to be stochastic. The stochasticity of the reference policy could be to account for the unmodeled elements of the dynamics, to provide robustness, or to encourage exploration.*

Remark 7. *Consider a special setting in which the state dynamics $F_t(x_t, u_t)$ is linear in x_t and u_t, the reference policy distribution $R_{U_t|X_t}(\cdot|x_t)$ is Gaussian, and the cost functions $C_t(\cdot,\cdot)$ and $C_T(\cdot)$ are quadratic in x_t and u_t. In such a setting, it can be shown that the optimal deceptive policy $Q^*_{U_t|X_t}(\cdot|x_t)$ is also Gaussian and can be analytically computed by backward Riccati recursions similar to the standard Linear-Quadratic-Regular (LQR) problems. In this work, we consider a broader setting with possibly non-Gaussian reference distribution, non-linear state dynamics, and non-quadratic cost functions. In this case, the optimal deceptive policy might not be efficiently computed by solving backward recursions.*

Now, we propose a path-integral-based solution approach for simulator-driven policy synthesis. Under assumption 5, we can rewrite (124) as

$$J_t(x_t) = -\lambda \log \left\{ \int_{\mathcal{U}_t} \exp\left(-\frac{C_t(x_t, u_t)}{\lambda}\right) \times \exp\left(-\frac{J_{t+1}\left(F_t(x_t, u_t)\right)}{\lambda}\right) R(du_t|x_t) \right\}.$$

We introduce the exponentiated value function as $Z_t(x_t) := \exp\left(-\frac{1}{\lambda}J_t(x_t)\right)$. Using $Z_t(x_t)$, the Bellman recursion (130) can be linearized, and we get the following linear relationship between Z_t and Z_{t+1}:

$$\begin{aligned} Z_t(x_t) &= \int_{\mathcal{U}_t} \exp\left(-\frac{C_t(x_t, u_t)}{\lambda}\right) Z_{t+1}\left(F_t(x_t, u_t)\right) R(du_t|x_t) \\ &= \int_{\mathcal{U}_t}\int_{\mathcal{X}_{t+1}} \exp\left(-\frac{C_t(x_t, u_t)}{\lambda}\right) Z_{t+1}(x_{t+1}) \times P(dx_{t+1}|x_t, u_t) R(du_t|x_t). \end{aligned} \tag{130}$$

Note that in (130), $P(dx_{t+1}|x_t, u_t) = \delta_{F_t(x_t,u_t)}(dx_{t+1})$ by Assumption 5. Equation (130) is a linear backward recursion in Z_t. The linear solvability of the KL control problem is well-known in the literature (e.g., [109]). We remark that linearizability critically relies on Assumption 5[4]. Now, by recursive substitution, (130) can also be written as

$$Z_t(x_t) = \int_{\mathcal{U}_t}\int_{\mathcal{X}_{t+1}} \cdots \int_{\mathcal{U}_{T-1}}\int_{\mathcal{X}_T} \exp\left(-\frac{C_t(x_t, u_t)}{\lambda}\right) \times \cdots \times \exp\left(-\frac{C_T(x_T)}{\lambda}\right) R(dx_{t+1:T} \times du_{t:T-1}|x_t).$$

Thus, by introducing the path cost function

$$C_{t:T}(x_{t:T}, u_{t:T-1}) := \sum_{k=t}^{T-1} C_k(x_k, u_k) + C_T(x_T), \tag{131}$$

we obtain

$$Z_t(x_t) = \mathbb{E}_R \exp\left(-\frac{1}{\lambda} C_{t:T}(X_{t:T}, U_{t:T-1})\right) \tag{132}$$

where the expectation $\mathbb{E}_R(\cdot)$ is with respect to the probability measure R (114). Equation (132) expresses the exponentiated value function $Z_t(x_t)$ as the expected path cost under the reference distribution. This suggests a path-integral-based

[4] We remark that in prior works where the path integral method is used to solve stochastic control problems, a certain assumption (e.g. Eq. (9) in [2]) is made to reinterpret the original problem as a problem of designing the optimal randomized policy for a deterministic transition system.

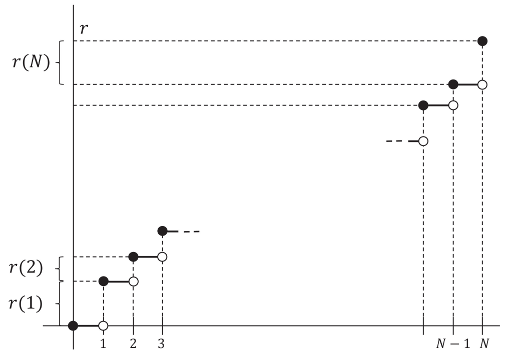


Figure 21: Function $F_t(x)$.

approach to numerically compute $Z_t(x_t)$. Suppose we generate a collection of N independent samples of system paths $\{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^N$ starting from x_t under the reference distribution R. Since the reference distribution is known, a collection of such sample paths can be easily generated using a Monte Carlo simulation. If $C_{t:T}(x_{t:T}(i), u_{t:T-1}(i))$ represents the path cost of the sample path i, then by the strong law of large numbers [1] as $N \to \infty$, we get

$$\frac{1}{N}\sum_{i=1}^{N} \exp\left(-\frac{1}{\lambda}C_{t:T}(x_{t:T}(i), u_{t:T-1}(i))\right) \overset{a.s.}{\to} Z_t(x_t). \tag{133}$$

Similarly, a collection of sample paths $\{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^N$ starting from x_t under the reference distribution R can be used to sample u_t from the optimal distribution $Q^*_{U_t|X_t}(\cdot|x_t)$. Notice that the optimal deceptive policy (124) can be expressed in terms of $\{Z_t\}_{t=0}^T$ as

$$\begin{aligned}
Q^*_{U_t|X_t}(B_{U_t}|x_t) =& \frac{1}{Z_t(x_t)}\int_{B_{U_t}} \exp\left(-\frac{C_t(x_t,u_t)}{\lambda}\right) \times \exp\left(-\frac{J_{t+1}\left(F_t(x_t,u_t)\right)}{\lambda}\right) R(du_t|x_t) \\
=& \frac{1}{Z_t(x_t)}\int_{B_{U_t}} \exp\left(-\frac{C_t(x_t,u_t)}{\lambda}\right) \times Z_{t+1}\left(F_t(x_t,u_t)\right) R(du_t|x_t) && \text{(134a)}\\
=& \frac{1}{Z_t(x_t)}\int_{B_{U_t}}\int_{\mathcal{X}_{t+1}} \exp\left(-\frac{C_t(x_t,u_t)}{\lambda}\right) Z_{t+1}(x_{t+1}) \times P(dx_{t+1}|x_t,u_t)R(du_t|x_t) && \text{(134b)}\\
=& \frac{1}{Z_t(x_t)}\int_{\{\mathcal{X}_{t+1:T},\mathcal{U}_{t:T-1}|u_t\in B_{U_t}\}} \exp\left(-\frac{C_{t:T}(x_{t:T},u_{t:T-1})}{\lambda}\right) \times R(dx_{t+1:T}\times du_{t:T-1}|x_t). && \text{(134c)}
\end{aligned}$$

The step (134a) follows from the definition of Z_t. In (134b), we used our assumption $P(dx_{t+1}|x_t,u_t) = \delta_{F_t(x_t,u_t)}(dx_{t+1})$. Finally, (134c) is obtained by the recursive substitution of (134b), and $\{\mathcal{X}_{t+1:T}, \mathcal{U}_{t:T-1}|u_t \in B_{U_t}\}$ represents a collection paths such that $u_t \in B_{U_t}$.

We use the above representation of $Q^*_{U_t|X_t}$ to sample an action u_t from it. Let $r_t(i)$ be the exponentiated path cost of the sample path i:

$$r_t(i) := \exp\left(-\frac{1}{\lambda}C_{t:T}(x_{t:T}(i), u_{t:T-1}(i))\right) \tag{135}$$

and $r_t := \sum_{i=1}^N r_t(i)$. For each $t \in \mathcal{T}$, we introduce a piecewise constant, monotonically non-decreasing function $F_t : [0,N] \to [0, r_t]$ by

$$F_t(x) = \sum_{i=1}^{\lfloor x \rfloor} r_t(i).$$

where $\lfloor x \rfloor$ denotes floor(x), i.e., the greatest integer less than or equal to x. The function $F_t(x)$ is represented in Figure 21.

Notice that the inverse F_t^{-1} of F_t defines a mapping $F_t^{-1} : [0, r_t] \to \{1, 2, ..., N\}$. To generate a sample u_t approximately from the optimal distribution $Q^*_{U_t|X_t}$, we propose Algorithm 2. We first, generate a random variable d according

Algorithm 2 Sampling u_t approximately from $Q^*_{U_t|X_t}(\cdot|x_t)$ by Monte Carlo simulations

```
Require: Initial state $x_0$
1: for $t \in \mathcal{T}$ do
2:     Sample $N$ paths $\{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^N$ starting from $x_t$ under the reference distribution $R$.
3:     For each sample path $i$, compute the exponentiated path cost $r_t(i)$ by (135).
4:     Compute $r_t := \sum_{i=1}^N r_t(i)$.
5:     Generate a random variable $d$ according to $d \sim \text{unif}[0, r_t]$.
6:     Select a sample ID by $j_t \leftarrow F_t^{-1}(d)$.
7:     Select a control input as $u_t \leftarrow u_t(j_t)$.
8: end for
```

to $d \sim \text{unif}[0, r_t]$. Then, we select a sample ID by $j_t \leftarrow F_t^{-1}(d)$. Finally, the control input adopted in the j_t-th sample path at time step t is selected as u_t, i.e., $u_t \leftarrow u_t(j_t)$. Theorem 13 proves that as the number of Monte Carlo samples tends to infinity, Algorithm 1 samples u_t from the optimal distribution $Q^*_{U_t|X_t}(\cdot|x_t)$.

Theorem 13. *Let $B_{U_t} \in \mathcal{B}(\mathcal{U}_t)$ be a Borel set. Suppose for a given collection of sample paths $\{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^N$, u_t is computed by Algorithm 1 and the probability of $u_t \in B_{U_t}$ is denoted by $\Pr\{u_t \in B_{U_t} | \{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^N\}$. Then, as $N \to \infty$*

$$\Pr\{u_t \in B_{U_t} | \{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^N\} \xrightarrow{a.s.} Q^*_{U_t|X_t}(B_{U_t}|x_t).$$

Proof. Let $\mathcal{I}_{B_{U_t}}$ be the set of indices of sample paths for which an action in B_{U_t} is taken at time step t, i.e.,

$$\mathcal{I}_{B_{U_t}} = \{i \in \{1, 2, \ldots, N\} | u_t(i) \in B_{U_t}\}.$$

Define the sum of the exponentiated path costs of the paths in $\mathcal{I}_{B_{U_t}}$ as

$$r_{B_{U_t}} = \sum_{i \in \mathcal{I}_{B_{U_t}}} r_t(i).$$

By construction of Algorithm 2,

$$\Pr\{u_t \in B_{U_t} | \{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^N\} = \frac{r_{B_{U_t}}}{r_t}. \tag{136}$$

Now, from (133), as $N \to \infty$, we get

$$\frac{r_t}{N} = \frac{1}{N} \sum_{i=1}^N \exp\left(-\frac{C_{t:T}(x_{t:T}(i), u_{t:T-1}(i))}{\lambda}\right) \xrightarrow{a.s.} Z_t(x_t).$$

Similarly, as $N \to \infty$,

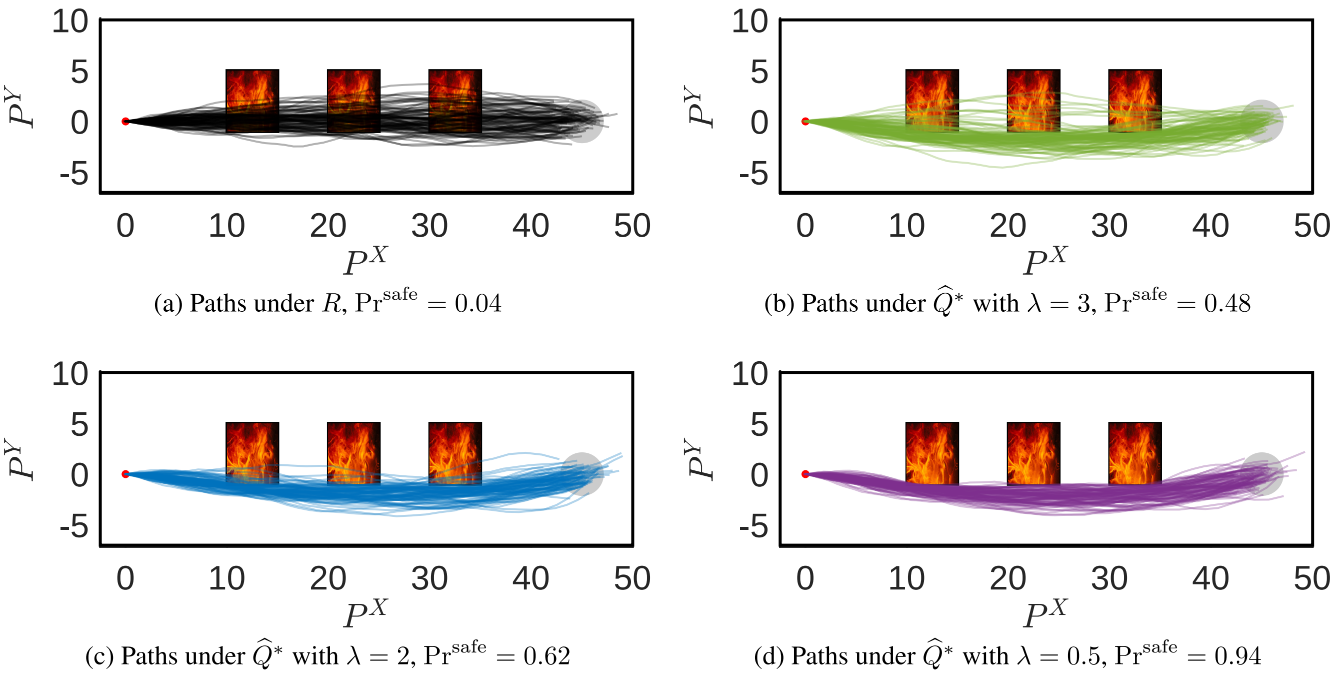
$$\begin{aligned}\frac{r_{B_{U_t}}}{N} =& \frac{1}{N} \sum_{i \in \mathcal{I}_{B_{U_t}}} \exp\left(-\frac{C_{t:T}(x_{t:T}(i), u_{t:T-1}(i))}{\lambda}\right) \\ &\xrightarrow{a.s.} \int_{\{\mathcal{X}_{t+1:T}, \mathcal{U}_{t:T-1} | u_t \in B_{U_t}\}} \exp\left(-\frac{C_{t:T}(x_{t:T}, u_{t:T-1})}{\lambda}\right) \times R(dx_{t+1:T}, du_{t:T-1}|x_t)\end{aligned}$$

Thus, from (134c) and (136)

$$\begin{aligned}\Pr\{u_t \in B_{U_t} | \{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^N\} &\xrightarrow{a.s.} \frac{1}{Z_t(x_t)} \int_{\{\mathcal{X}_{t+1:T}, \mathcal{U}_{t:T-1} | u_t \in B_{U_t}\}} \exp\left(-\frac{C_{t:T}(x_{t:T}, u_{t:T-1})}{\lambda}\right) \times R(dx_{t+1:T}, du_{t:T-1}|x_t) \\ &= Q^*_{U_t|X_t}(B_{U_t}|x_t).\end{aligned}$$

□

We showed that under Assumption 5, the optimal deceptive policies can be synthesized using path integral control. Algorithm 2 allows the deceptive agent to numerically compute optimal actions via Monte Carlo simulations without explicitly synthesizing the policy. Since Monte Carlo simulations can be efficiently parallelized, the agent can generate the optimal control actions online.



(a) Paths under R, $\mathrm{Pr}^{\mathrm{safe}} = 0.04$

(b) Paths under $\widehat{Q}^*$ with $\lambda = 3$, $\mathrm{Pr}^{\mathrm{safe}} = 0.48$

(c) Paths under $\widehat{Q}^*$ with $\lambda = 2$, $\mathrm{Pr}^{\mathrm{safe}} = 0.62$

(d) Paths under $\widehat{Q}^*$ with $\lambda = 0.5$, $\mathrm{Pr}^{\mathrm{safe}} = 0.94$

Figure 22: A unicycle navigation problem. The start position is shown by a red dot, and the goal region by a disk colored in gray. 100 sample paths generated under the reference policy R and the agent's policy $\widehat{Q}^*$ with three values of λ are shown. The probability of safe paths $\mathrm{Pr}^{\mathrm{safe}}$ are noted below each case.

5.6 Simulation Results

In this section, we validate the path-integral-based algorithm proposed to generate optimal deceptive control actions. The problem is illustrated in Figure 22. A supervisor wants an agent to start from the origin and reach a disk of radius G^R centered at $\begin{bmatrix} G^X & G^Y \end{bmatrix}^\top$ (shown in gray color) as fast as possible. The supervisor also expects the agent to inspect the region on the way. To encourage exploration and to provide robustness against unmodeled dynamics, the supervisor designs a randomized reference policy. The agent, on the other hand, wishes to avoid the regions on the way that are covered under fire, as shown in Figure 22. Let these regions be represented collectively by $\mathcal{X}^{\mathrm{fire}}$. Suppose the agent's dynamics are modeled by a unicycle model as:

$$P^X_{t+1} = P^X_t + S_t \cos \Theta_t h, \quad P^Y_{t+1} = P^Y_t + S_t \sin \Theta_t h, \quad S_{t+1} = S_t + A_t h, \quad \Theta_{t+1} = \Theta_t + \Omega_t h$$

where (P^X_t, P^Y_t), S_t, and Θ_t denote the $x-y$ position, speed, and the heading angle of the agent at time step t, respectively. The control input $U_t := \begin{bmatrix} A_t & \Omega_t \end{bmatrix}^T$ consists of acceleration A_t and angular speed Ω_t. h is the time discretization parameter used for discretizing the continuous-time unicycle model. For this simulation study, we set $h = 1$. Note that the agent's dynamics is deterministic as per Assumption 5; however, the control input U_t can be stochastic. Suppose the supervisor designs the reference policy R as a Gaussian probability density with mean $\overline{u}_t$ and covariance Σ_t:

$$R_{U_t|X_t}(\cdot|x_t) = \frac{\exp\left[-\frac{1}{2}(u_t - \overline{u}_t)^\top \Sigma_t^{-1}(u_t - \overline{u}_t)\right]}{\sqrt{(2\pi)^2 |\Sigma_t|}}.$$

The mean $\overline{u}_t := \begin{bmatrix} \overline{a}_t & \overline{\omega}_t \end{bmatrix}^\top$ is designed using a proportional controller as

$$\overline{A}_t = -k_A(S_t - S_t^{\mathrm{desired}}), \quad \overline{\Omega}_t = -k_\Omega(\Theta_t - \Theta_t^{\mathrm{desired}})$$

where k_A and k_Ω are proportional gains and S_t^{desired}, $\Theta_t^{\mathrm{desired}}$ are computed as

$$S_t^{\mathrm{desired}} = \frac{\left\| \begin{bmatrix} G^X \\ G^Y \end{bmatrix} - \begin{bmatrix} P^X_t \\ P^Y_t \end{bmatrix} \right\|}{T - t}, \quad \Theta_t^{\mathrm{desired}} = \tan^{-1}\left(\frac{G^Y - P^Y_t}{G^X - P^X_t} \right).$$

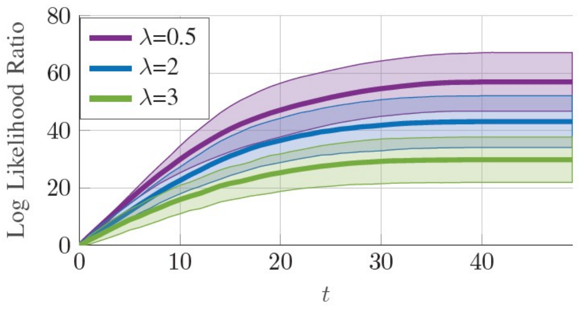


Figure 23: Expected LLR (with one standard deviation) with respect to time t for three values of λ.

As mentioned before, the agent wishes to avoid the region $\mathcal{X}^{\text{fire}}$. Suppose the cost function $C_{0:T}$ is designed as

$$C_{0:T}(X_{0:T}, U_{0:T-1}) = \sum_{t=0}^{T} \mathbb{1}_{[P_t^X \ P_t^Y]^\top \in \mathcal{X}^{\text{fire}}}$$

where $\mathbb{1}_{[P_t^X \ P_t^Y]^\top \in \mathcal{X}^{\text{fire}}}$ represents an indicator function that returns 1 when the agent is inside the region $\mathcal{X}^{\text{fire}}$ and 0 otherwise. For this simulation, we set

$$\begin{bmatrix} G^X \\ G^Y \end{bmatrix} = \begin{bmatrix} 45 \\ 0 \end{bmatrix}, \quad \Sigma_t = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \quad k_A = 0.1, \quad k_\Omega = 0.2, \quad T = 50.$$

The agent chooses its action at each time step using Algorithm 2 where the number of samples is $N = 10^5$.

Suppose $\widehat{Q}^*$ denotes the deceptive agent's distribution generated by the sampling-based Algorithm 1. Figure 22 shows 100 paths under the reference distribution R (Figure 22(a)) and the agent's distribution $\widehat{Q}^*$ for three values of λ (Figure 22(b) - 22(d)). A lower value of λ implies that the agent cares less about its deviation from the reference policy and more about avoiding the region $\mathcal{X}^{\text{fire}}$. A higher value of λ implies the opposite. We also report $\Pr^{\text{safe}}$, the percentage of paths that avoid $\mathcal{X}^{\text{fire}}$. Under the reference distribution R, only 4% of the paths are safe. On the other hand, more paths are safe under the agent's distribution $\widehat{Q}^*$, and as the value of λ reduces, $\Pr^{\text{safe}}$ increases.

Figure 23 shows the expected log-likelihood ratio (with one standard deviation) with respect to time t for three values of λ. The expected LLR is computed as follows. Algorithm 2 selects a control input $u_k \leftarrow u_k(j_k)$ at time step k, where j_k is a sample ID obtained from step 5. From the construction of the algorithm, at each time step k, the probability of choosing the control input $u_k \leftarrow u_k(j_k)$ under the agent's distribution $\widehat{Q}^*$ is $r_k(j_k)/r_k$, where $r_k(j_k)$ and r_k are computed by steps 3 and 4 of Algorithm 2. Whereas the probability of choosing the control input $u_k \leftarrow u_k(j_k)$ under the reference distribution is $1/N$. Therefore, using (120), the expected LLR upto time $t \in \mathcal{T}$ can be approximately computed as

$$\mathbb{E}_{Q^*}\left[\log \frac{dQ^*_{X_{0:t}\times U_{0:t-1}}}{dR_{X_{0:t}\times U_{0:t-1}}}(x_{0:t}, u_{0:t-1})\right] = \mathbb{E}_{Q^*}\left[\sum_{k=0}^{t-1} \log \frac{dQ^*_{U_k|X_k}}{dR_{U_k|X_k}}(x_k, u_k)\right] \approx \frac{1}{N_{\widehat{Q}^*}} \sum_{i=1}^{N_{\widehat{Q}^*}} \sum_{k=0}^{t-1} \frac{r_k(j_k)/r_k}{1/N}.$$

where $N_{\widehat{Q}^*}$ is the number of paths generated by repeatedly running Algorithm 2. Note that since we assume the system dynamics to be deterministic (Assumption 5), once the control input u_k is chosen at time step k, the state x_{k+1} is uniquely determined. Therefore, while computing the expected LLR, we only need to consider the probabilities of choosing the control input u_k under policies $\widehat{Q}^*$ and R. Figure 23 shows that for a lower value of λ, the expected LLR is higher, i.e., more deviation of $\widehat{Q}^*$ from R.

5.7 Publications

- **A. Patil**, M. Karabag, U. Topcu, T. Tanaka, "Simulation-Driven Deceptive Control via Path Integral Approach," *2023 IEEE Conference on Decision and Control (CDC)*

5.8 Future Work

For future work, we plan to study the deception problem for continuous-time stochastic systems. We also plan to conduct the sample complexity analysis of the path integral approach to solve KL control problems.

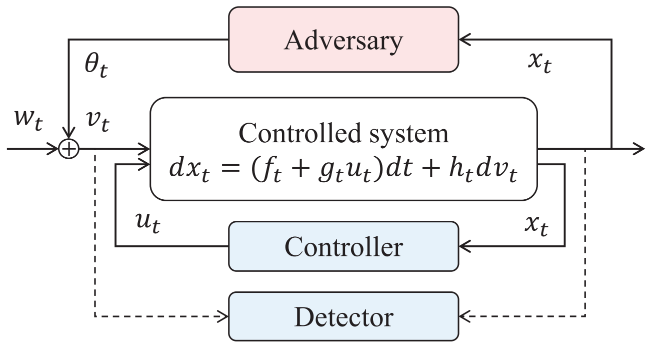


Figure 24: Attacker vs controller/detector.

6 Detection and Risk Mitigation of Stealthy Attack

In the remaining 6 months of the PhD program, we propose to solve another class of SOC problem using path integral approach, namely "Detection and Risk Mitigation of Stealthy Attack". This work would be an extension of our work in deceptive control for discrete-time stochastic systems presented in Section 5 and the two-player zero-sum stochastic differential game presented in Section 3.

Consider a system shown in Figure 24 following nonlinear dynamics with state $x(t) \in \mathbb{R}^n$, $0 \le t \le T$:

$$dx(t) = f\left(x(t)\right)dt + G\left(x(t)\right)u(x(t))dt + H\left(x(t), t\right)dv(t), \quad x(0) = x_0 \tag{137a}$$
$$dv(t) = \theta(t)dt + dw(t). \tag{137b}$$

There are two players competing on this state space model. The first input $u(t) \in \mathbb{R}^\ell$, $0 \le t \le T$ is determined by the controller who adopts a state feedback policy (i.e., $u(\cdot)$ is measurable with respect to the σ-algebra generated by $x(s), 0 \le s \le t$). The second input $\theta(t) \in \mathbb{R}^m$, $0 \le t \le T$ is determined by the adversary who also adopts a state feedback policy (i.e., $\theta(\cdot)$ measurable with respect to the σ-algebra generated by $x(s), 0 \le s \le t$). The noise process $w(t)$, $0 \le t \le T$ is an m-dimensional standard Brownian motion. Assume that the adversary injects an attack signal $\theta(t)$ in the system and the controller is able to observe $v(t)$ (but not $\theta(t)$) to detect the presence of the attack signal. When the attack signal is absent, the system input $v(t)$ is equal to the standard Brownian motion $w(t)$. That is,

$$dv(t) = dw(t) \quad \text{(No attack)}$$
$$dv(t) = \theta(t)dt + dw(t) \quad \text{(Under attack)}.$$

Attack detection can be performed by monitoring the statistical deviation of $v(t)$ from the standard Brownian motion. For example, prior works [106, 105, 112, 113] suggested to use the KL divergence $D(P\|Q)$, where P is the probability measure in which $w(t)$ is the standard Brownian motion and Q is the measure in which $v(t)$ is the Brownian motion. Adopting $D(P\|Q)$ as the stealthiness measure, the adversary is incentivized to keep $D(P\|Q)$ small.

6.1 Worst-Case Attack Synthesis

We define the following problem as the attacker's problem to synthesize worst-case attack

$$\max_{\theta(\cdot)} \mathbb{E}^P \left[\int_0^T C(x(t), u(t))dt + C(x(T)) \right] \tag{138}$$
$$s.t. \;\; D(P\|Q) \le \epsilon$$

We construct a dual of the optimal control problem (138), show that strong duality holds and establish an equivalence between the "hard-constrained" problem (138) and the "soft-constrained" problem:

$$\max_{\theta(\cdot)} \; \mathbb{E}^P \left[\int_0^T C(x(t), u(t))dt + C(x(T)) \right] - \lambda D(P\|Q) \tag{139}$$

where $\lambda \ge 0$ is a fixed constant. We show that the path integral method is applicable to solve the continuous-time KL-control problem (139). Specifically, if the attacker has a simulator engine that can generate a large number of sample trajectories $\{x(t)^i\}_{i=1}^N$ from the distribution Q starting from the current state-time pair (x, t), then the optimal attack signal $\theta(t)$ can be computed directly from the sample ensemble $\{x(t)^i\}_{i=1}^N$.

6.2 Attack risk mitigation

In this section, we turn our attention to the system operator's problem, who is interested in mitigating the risk of stealthy attacks. We consider the scenario shown in Figure 24 where the system operator is now able to apply a control input $u(t)$ to combat with the potential attack input $\theta(t)$. We assume the existence of an attack detector, who tests the hypothesis H_0 (no attack in progress) against the alternative H_1 (attack in progress) based on the observed input $0 \leq v(t) \leq T$. We assume the Neyman-Pearson test based on the likelihood ratio $\frac{dP}{dQ}$ is adopted. We model the competition between the operator and the attacker as a minimax game in which the operator acts as the minimizer and the attacker acts as the maximize. The problem we consider in this section is

$$\min_{u(\cdot)} \max_{\theta(\cdot)} \mathbb{E}^P \left[\int_0^T C(x(t), u(t))dt + C(x(T)) \right] - \lambda D(P\|Q) \tag{140}$$

The KL divergence term $D(P\|Q)$, which does not depend on the minimizer's policy, serves as the stealthiness measure of the attacker's policy.

6.3 Experimental Results

In this research, we will perform an extensive series of simulation studies using the *Isaac Sim* simulator, a powerful tool widely used for testing robotic algorithms in virtual environments. Isaac Sim allows for high-fidelity simulation of various robotic platforms, enabling us to rigorously evaluate the performance of the proposed control strategies in a controlled and repeatable manner. Through these simulations, we aim to validate the effectiveness of our path integral control framework in detecting and mitigating the stealthy attacks.

In addition to the simulated experiments, we plan to conduct real-world tests to further demonstrate the practical applicability of our approach. These tests will be performed on physical robotic platforms, such as the *Qbot* mobile robot developed by Quanser and the quadruped robot, *Spot*, developed by Boston Dynamics. The Qbot will allow us to explore scenarios involving ground-based navigation and obstacle avoidance, while Spot will be used for more advanced experiments involving dynamic locomotion and real-time adaptation to unpredictable environments.

By combining simulation studies with real-world experiments, we aim to provide comprehensive evidence of the robustness and adaptability of the proposed control strategies, ensuring their viability for deployment in real-world autonomous systems.

7 Discrete-Time Stochastic LQR via Path Integral Control and Its Sample Complexity Analysis

7.1 Motivation and Literature Review

We presented five application domains of path integral control approach. Although the path integral method is convenient for a broad class of stochastic optimal control problems, the outcomes of Monte-Carlo simulations are inherently probabilistic; therefore, applying path integral methods to safety-critical missions would require rigorous performance guarantees. The purpose of this section is to introduce a theoretical framework for sample complexity analysis that allows us to understand the interplay between the achievable control performance by the path integral method and the required computational cost.

Despite the wide applicability of path integral control, not enough work has been done on its sample complexity analysis. To the best of the authors' knowledge, the only prior work addressing this problem is the work by Yoon et al. [66]. The authors of [66] considered the continuous-time path integral control, and applied Chebyshev and Hoeffding inequalities to relate the instantaneous (pointwise-in-time) error bound in control input and the sample size of the Monte-Carlo simulations performed at that particular time instance. While this result sheds light on the sample complexity analysis, the work [66] is limited in the following aspects. First, the effect of time discretization is not addressed. Since numerical implementations of path integral methods necessitate time discretization, and the control performance depends on the discretization step sizes, such an analysis is not negligible. Second, it is not clear how the pointwise-in-time bound in [66] can be translated into a more explicit, end-to-end (trajectory-level) error bound. Third, [66] does not provide machinery to compute the required sample size to achieve an acceptable loss of control performance. Such an analysis is crucial for safety-critical applications.

7.2 Contributions

We make the following contributions to this work:

1. Derivation of a path integral formulation for discrete-time stochastic Linear Quadratic Regulator (LQR) problem: Recognizing the difficulties in answering the above questions in the continuous-time setting, we take a step back and study the sample complexity of path integral in the discrete-time setting. We develop a discrete-time counterpart of the path integral control scheme for the stochastic LQR problems. Since a formulation of path integral control for discrete-time LQR has not been provided explicitly in the literature (note, however, a recent work [114]), we decided to derive it based on a more general setup of the so-called Kullback-Leibler (KL) control problem and using its intimate connection with path integral control [34, 115].

2. Derivation of an end-to-end (trajectory-level) bound on the error in the control signals: There is often a need to assess the safety of the control policy across the entire time horizon. Notably, the end-to-end error bound in the control signals is challenging to evaluate because it couples states across the time horizon as a whole. In this work, we provide the end-to-end bound on the error between the optimal control signal (computed by the classical Riccati solution) and the one obtained by the path integral method as a function of sample sizes. Our sample complexity analysis reveals that the required number of samples for path integral control depends only logarithmically on the dimension of the control input.

3. Formulation of a chance-constrained optimization problem to quantify the worst-case performance of the path integral LQR control: We provide a method to quantify the worst-case control performance by formulating a chance-constrained LQR problem for an adversarial agent. This result, together with (2), will allow us to relate the sample size and the worst-case control performance of the path integral method.

The main purpose of our study is to establish a theoretical foundation for the sample complexity of the path integral method. While the path integral method is particularly powerful in nonlinear settings, in this work, we have chosen the LQR setup since the availability of an analytical expression of the optimal control law allows for quantitative error analysis. The results presented in this work will be the necessary first step for future study on the sample complexity of the path integral method for nonlinear systems.

Notations

We use the same notations as described in Section 5.

7.3 KL Control via Path Integral

In this section, we introduce a class of discrete-time continuous-state KL control problems similar to Section 5.4 and present the path-integral-based solution approach. Let $\mathcal{X}_t \subseteq \mathbb{R}^n$ and $\mathcal{U}_t \subseteq \mathbb{R}^m$ be the spaces of states and control inputs at time step t, respectively. Suppose the state transition is governed by the mapping $x_{t+1} = F_t(x_t, u_t)$, $F_t : \mathcal{X}_t \times \mathcal{U}_t \to \mathcal{X}_{t+1}$ for each $t \in \mathcal{T} := \{0, 1, \cdots, T\}$ with a given initial state $X_0 = x_0$. Notice that we assume the state transition law is deterministic and can also be written as $P_{X_{t+1}|X_t,U_t}(dx_{t+1}|x_t, u_t) = \delta_{F_t(x_t,u_t)}(dx_{t+1})$ where δ is the Dirac measure. In contrast, the control policy to be designed is allowed to be randomized and is represented by a sequence of stochastic kernels [116] $Q_{U_t|X_t}, t \in \mathcal{T}$. A sequence of stochastic kernels $R_{U_t|X_t}, t \in \mathcal{T}$ represents a nominal (reference) policy which is given in advance. We denote the joint probability distributions of the state-control trajectories induced by the policies R and Q as Equations (114) and (115) respectively. Let the functions $C_t(\cdot, \cdot) : \mathcal{X}_t \times \mathcal{U}_t \to \mathbb{R}$ for $t \in \mathcal{T}$ and $C_T(\cdot) : \mathcal{X}_T \to \mathbb{R}$ represent the stage-wise and the terminal cost functions, respectively and the path cost can be written as (131). Introducing a positive weighing factor λ, the KL control problem is formulated as:

Problem 6 (KL control problem)**.**

$$\min_{\{Q_{U_t|X_t}\}_{t=0}^{T-1}} \quad \mathbb{E}_Q^{x_0} \, C_{0:T}(X_{0:T}, U_{0:T-1}) + \lambda D(Q_{X_{0:T},U_{0:T-1}} \| R_{X_{0:T},U_{0:T-1}}). \tag{141}$$

The expectation $\mathbb{E}_Q^{x_0}(\cdot)$ is with respect to the probability measure (115) with a fixed initial state x_0. λ is a positive constant that balances a trade-off between the path cost $C_{0:T}(X_{0:T}, U_{0:T-1})$ and the KL divergence $D(Q_{X_{0:T},U_{0:T-1}} \| R_{X_{0:T},U_{0:T-1}})$. Define the value function for Problem 6 by

$$J_t(x_t) := \min_{\{Q_{U_k|X_k}\}_{k=t}^{T-1}} \mathbb{E}_Q^{x_t}\, C_{t:T}(X_{t:T}, U_{t:T-1}) + \lambda D(Q_{X_{t:T},U_{t:T-1}} \| R_{X_{t:T},U_{t:T-1}}).$$

Here, $\mathbb{E}_Q^{x_t}$ denotes the expectation with respect to the measure $Q_{X_{t:T},U_{t:T-1}}$ induced by the policy $\{Q_{U_k|X_k}\}_{k=t}^{T-1}$ and the state transition law $x_{k+1} = F_k(x_k, u_k)$, $k = t, \cdots T-1$ with a fixed initial state x_t. The expectation $\mathbb{E}_R^{x_t}$ with respect to the measure $R_{X_{t:T},U_{t:T-1}}$ is defined similarly. The next theorem provides an explicit representation of the value function, which plays a central role in the path integral control algorithm. This result can be thought of as a discrete-time counterpart of the Feynmann-Kac lemma used in [2] in the derivation of continuous-time path integral control.

Theorem 14. *For each $t \in \mathcal{T}$, the value function admits a representation*

$$J_t(x_t) = -\lambda \log \mathbb{E}_R^{x_t} \exp\left(-\frac{1}{\lambda} C_{t:T}(X_{t:T}, U_{t:T-1})\right). \tag{142}$$

The optimal policy $Q^*_{U_t|X_t}$ for Problem 6 is expressed as (124).

Proof. The proof follows similar to the proof of Theorem 12. □

Equation (142) implies that the value function $J_t(x_t)$ can be computed approximately using independent Monte-Carlo simulations of state-control trajectories starting from x_t under the reference policy $\{R_{U_k|X_k}\}_{k=t}^{T-1}$. Let $\{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^n$ be an ensemble of n such sample state-control trajectories. For each i, let

$$r(i) := \exp\left(-\frac{1}{\lambda} C_{t:T}(x_{t:T}(i), u_{t:T-1}(i))\right) \tag{143}$$

be the reward of the sample path i. Then, $J_t(x_t)$ can be evaluated approximately by $J_t(x_t) \approx -\lambda \log\left(\frac{1}{n}\sum_{i=1}^n r(i)\right)$. Moreover, the expectation of the control input under the optimal distribution $Q^*_{U_t|X_t}$ (124) can also be evaluated as

$$\begin{aligned}
\mathbb{E}_{Q^*}(U_t|x_t) &= \int_{\mathcal{U}_t} u_t Q^*(du_t|x_t) \\
&= \frac{\mathbb{E}_R\left[U_t \exp\left(-\frac{1}{\lambda} C_{t:T}(X_{t:T}, U_{t:T-1})\right)\right]}{\mathbb{E}_R\left[\exp\left(-\frac{1}{\lambda} C_{t:T}(X_{t:T}, U_{t:T-1})\right)\right]} \\
&\approx \frac{\sum_{i=1}^n u_t(i) r(i)}{\sum_{i=1}^n r(i)}.
\end{aligned} \tag{144}$$

7.4 Stochastic LQR via Path Integral

In this section, we obtain the path integral algorithm for discrete-time stochastic LQR as a special case of the discrete-time, continuous-state KL control problem presented in Section 5.4. Suppose for each $t \in \mathcal{T}$, the state transition is governed by the following linear difference equation

$$X_{t+1} = A_t X_t + B_t U_t + W_t \tag{145}$$

where $X_t \in \mathbb{R}^n$, $U_t \in \mathbb{R}^m$ and $W_t \sim \mathcal{N}(0, \Omega_t), t \in \mathcal{T}$ are mutually independent Gaussian random variables. We assume that the initial state $X_0 = x_0$ is given. Now we state the stochastic LQR problem as follows:

Problem 7 (Stochastic LQR)**.** *Compute the state feedback policy $u_t = k_t(x_t)$ that solves the following optimal control problem:*

$$\begin{aligned}
&\min_{\{k_t(\cdot)\}_{t=0}^{T-1}} \mathbb{E} \sum_{t=0}^{T-1}\left(\frac{1}{2} X_t^\top M_t X_t + \frac{1}{2} U_t^\top N_t U_t\right) + \mathbb{E}\left(\frac{1}{2} X_T^\top M_T X_T\right) \\
&\text{s.t. } X_{t+1} = A_t X_t + B_t U_t + W_t, \quad X_0 = x_0
\end{aligned} \tag{146}$$

where $\{M_t\}_{t=0}^T$ and $\{N_t\}_{t=0}^{T-1}$ are sequences of positive definite matrices.

7.4.1 Classical solution

It is well-known that the optimal policy is given by

$$u_t = k_t(x_t) = K_t x_t, \tag{147a}$$
$$K_t = -(B_t^\top \Theta_{t+1} B_t + N_t)^{-1} B_t^\top \Theta_{t+1} A_t \tag{147b}$$

where $\{\Theta_t\}_{t=0}^T$ is a sequence of positive definite matrices computed by the backward Riccati recursion with $\Theta_T = M_T$:

$$\Theta_t = A_t^\top \Theta_{t+1} A_t + M_t - A_t^\top \Theta_{t+1} B_t (B_t^\top \Theta_{t+1} B_t + N_t)^{-1} B_t^\top \Theta_{t+1} A_t. \tag{148}$$

7.4.2 Path-integral-based solution

We now recover the classical result in Section 7.4.1 using the KL control framework described in Section 5.4. It turns out that Problem 2 can be related to Problem 1 under the following assumption:

Assumption 6. *For each $t = 0, 1, \cdots, T-1$, there exists a positive definite matrix $\hat{\Omega}_t$ and $\lambda > 0$ such that*

$$N_t = \lambda \hat{\Omega}_t^{-1} \text{ and } B_t \hat{\Omega}_t B_t^\top = \Omega_t. \tag{149}$$

This assumption implies that the control input in the direction with higher noise variance is cheaper than that in the direction with lower noise variance. In order to satisfy this assumption the noise W_t has to enter the dynamics via the control channel. While Assumption 6 is somewhat restrictive, it is a common assumption in the path integral control literature. See, e.g., [2] for its system theoretic interpretation. Its relaxations have been studied in the literature (e.g., [36]). For simplicity, we accept Assumption 1 in what follows.

Consider the KL control problem in which the state transition law is given by $x_{t+1} = F_t(x_t, u_t)$ where

$$F_t(x_t, u_t) = A_t x_t + B_t u_t \quad \forall t = 0, 1, \cdots, T-1. \tag{150}$$

Suppose that the reference policy $R_{U_t|X_t}$ is given as a non-degenerate zero-mean Gaussian probability density function with covariance $\hat{\Omega}_t$, i.e.,

$$R_{U_t|X_t}(u_t|x_t) = \mathcal{N}(0, \hat{\Omega}_t). \tag{151}$$

Finally, suppose that the stage-wise and terminal cost functions are given by

$$C_t(x_t, u_t) = \frac{1}{2} x_t^\top M_t x_t, \quad \forall t = 0, 1, \cdots, T-1 \tag{152}$$
$$C_T(x_T) = \frac{1}{2} x_T^\top M_T x_T. \tag{153}$$

Under this setup, Theorem 14 leads to the following result:

Theorem 15. *The KL control problem defined by (149) through (153) admits the optimal policy $Q^*_{U_t|X_t}$ given by*

$$Q^*_{U_t|X_t}(u_t|x_t) = \mathcal{N}(-\hat{H}_t^{-1} \hat{G}_t^\top x_t, \lambda \hat{H}_t^{-1}) \tag{154}$$

where $\hat{G}_t = A_t^\top \hat{\Theta}_{t+1} B_t$ and $\hat{H}_t = B_t^\top \hat{\Theta}_{t+1} B_t + \lambda \hat{\Omega}_t^{-1}$ and $\hat{\Theta}_t$ is the solution of the Riccati difference equation

$$\hat{\Theta}_t = A_t^\top \hat{\Theta}_{t+1} A_t + M_t - A_t^\top \hat{\Theta}_{t+1} B_t (B_t^\top \hat{\Theta}_{t+1} B_t + \lambda \hat{\Omega}_t^{-1})^{-1} B_t^\top \hat{\Theta}_{t+1} A_t \tag{155}$$

with $\hat{\Theta}_T = M_T$.

Proof. Suppose the value function $J_t(x_t)$ takes the form $J_t(x_t) = \frac{1}{2} x_t^\top \hat{\Theta}_t x_t + \hat{\kappa}_t$, where $\hat{\Theta}_T = M_T$ and $\hat{\kappa}_T = 0$. Define $Z_t(x_t) := \exp\left(-\frac{1}{\lambda} J_t(x_t)\right)$. Now, from (124), the optimal policy is given by

$$Q^*_{U_t|X_t}(u_t|x_t) = \frac{1}{Z_t(x_t)} \exp\left(-\frac{C_t(x_t, u_t)}{\lambda}\right) \times Z_{t+1}\left(F_t(x_t, u_t)\right) R(u_t|x_t). \tag{156}$$

Introducing $\hat{F}_t = A_t^\top \hat{\Theta}_{t+1} A_t + M_t$, $\hat{G}_t = A_t^\top \hat{\Theta}_{t+1} B_t$ and $\hat{H}_t = B_t^\top \hat{\Theta}_{t+1} B_t + \lambda \hat{\Omega}_t^{-1}$ and using the reference policy (151), equation (156) can be written as

$$Q^*_{U_t|X_t}(u_t|x_t) = \frac{1}{Z_t(x_t)} \frac{1}{\sqrt{(2\pi)^m \det \hat{\Omega}_t}} \times \exp\left(\frac{1}{2\lambda} x_t^\top (\hat{G}_t \hat{H}_t^{-1} \hat{G}_t^\top - \hat{F}_t) x_t - \frac{1}{\lambda} \hat{\kappa}_{t+1}\right)$$
$$\times \exp\left(-\frac{1}{2\lambda}(u_t + \hat{H}_t^{-1}\hat{G}_t^\top x_t)^\top \hat{H}_t (u_t + \hat{H}_t^{-1}\hat{G}_t^\top x_t)\right).$$

or (154) for short. From the condition $\int_{\mathcal{U}_t} Q^*_{U_t|X_t}(u_t|x_t) du_t = 1$, $Z_t(x_t)$ is determined as

$$Z_t(x_t) = \sqrt{\frac{\det\left(\lambda \hat{H}_t^{-1}\right)}{\det \hat{\Omega}_t}} \exp\left(\frac{x_t^\top (\hat{G}_t \hat{H}_t^{-1} \hat{G}_t^\top - \hat{F}_t) x_t}{2\lambda} - \frac{\hat{\kappa}_{t+1}}{\lambda}\right).$$

Therefore, the value function is written as

$$J_t(x_t) = -\lambda \log Z_t(x_t) = \frac{1}{2} x_t^\top \hat{\Theta}_t x_t + \hat{\kappa}_t$$

where $\hat{\Theta}_t$ is defined by (155) and $\hat{\kappa}_t$ by

$$\hat{\kappa}_t = \frac{\lambda}{2} \log \det \hat{\Omega}_t + \frac{\lambda}{2} \log \det\left(\hat{\Omega}_t^{-1} + \frac{1}{\lambda} B_t^\top \hat{\Theta}_{t+1} B_t\right) + \hat{\kappa}_{t+1}.$$

□

Critically, observe that Riccati equations (148) and (155) coincide under Assumption 6. Moreover, the mean of the control input under the optimal policy (154) is

$$\mathbb{E}_{Q^*}(u_t|x_t) = -\hat{H}_t^{-1} \hat{G}_t^\top x_t = K_t x_t, \tag{157}$$

which coincides with the LQR solution (147). Since (144) provides us with a Monte-Carlo-based approach to compute $\mathbb{E}_{Q^*}(u_t|x_t)$, this connection implies that the optimal LQR input (147) can be computed by Monte-Carlo simulations. Specifically, at each time step t, we generate sample state-control trajectories $\{x_{t:T}(i), u_{t:T-1}(i)\}_{i=1}^n$ under the reference policy $R_{U_k|X_k}$. Since the reference policy is currently given by (151), from Assumption 6, this amounts to performing n independent simulations of the "uncontrolled" dynamics $X_{k+1} = A_k X_k + W_k,\ W_k \sim \mathcal{N}(0, \Omega_k)$ for $k = t, \cdots, T-1$ with the initial state $X_t = x_t$. The path reward $r(i)$ is computed for each sample path using (143). Finally, the optimal control input for stochastic LQR is determined by

$$u_t \approx \frac{\sum_{i=1}^n r(i) u_t(i)}{\sum_{i=1}^n r(i)}. \tag{158}$$

Equation (158) is the discrete-time counterpart of the path integral control scheme introduced in the original work [2]. Notice that the path-integral-based LQR implementation using (158) does not require solving the backward Riccati equation (147).

7.5 Sample Complexity Analysis

Suppose $u_t = K_t x_t$ represents the optimal control input computed by solving the classical Riccati equation (148) and $\hat{u}_t$ represents the control input obtained via path integral approach using a finite number of samples $n_t \in \mathbb{N}$, i.e.,

$$\hat{u}_t = \sum_{i=1}^{n_t} \frac{r(i)}{r} u_t(i) \text{ where } r = \sum_{i=1}^{n_t} r(i). \tag{159}$$

7.5.1 Sample Complexity Bound

By the strong law of large numbers, $\hat{u}_t \overset{a.s.}{\to} u_t$, as $n_t \to \infty$. In this section, we analyze the finite sample performance of this approximation. Define the empirical means $\hat{E}$ and true expectations E as

$$\hat{E}_t^{ru} := \frac{1}{n_t}\sum_{i=1}^{n_t} r(i)u_t(i), \quad \hat{E}_t^{r} := \frac{1}{n_t}\sum_{i=1}^{n_t} r(i), \quad E_t^{ru} := \lim_{n_t\to\infty}\frac{1}{n_t}\sum_{i=1}^{n_t} r(i)u_t(i), \quad E_t^{r} := \lim_{n_t\to\infty}\frac{1}{n_t}\sum_{i=1}^{n_t} r(i). \tag{160}$$

Since $r(i) \in [0,1]$, by Hoeffding's inequality [117], we get

$$\Pr\left(\left|\hat{E}_t^r - E_t^r\right| \le \gamma_1\right) \ge 1 - 2e^{-2n_t\gamma_1^2}. \tag{161}$$

We know that $u_t(i)$ is sampled from $R_{U_t|X_t}$ in (151) which is a zero-mean Gaussian distribution with covariance $\hat{\Omega}_t$. Therefore, $u_t(i) \in \mathrm{SG}(\|\hat{\Omega}_t\|)$, where $\mathrm{SG}(\|\hat{\Omega}_t\|)$ represents a class of sub-Gaussian distributions with parameter $\|\hat{\Omega}_t\|$. Since $r(i) \in [0,1]$, we also have $(r(i)u_t(i)) \in \mathrm{SG}(\|\hat{\Omega}_t\|)$. Using Hoeffding's bound for sub-Gaussians and the union bound [118], we get

$$\Pr\left(\left\|\hat{E}_t^{ru} - E_t^{ru}\right\|_\infty \le \gamma_2\right) \ge 1 - 2me^{-n_t\gamma_2^2/(2\|\hat{\Omega}_t\|)} \tag{162}$$

where m is the dimension of $u_t(i)$. The following theorem provides an end-to-end sample complexity bound on the error in the path integral control signal.

Theorem 16. *Let $\{\epsilon_t\}_{t=0}^{T-1}$, $\{\alpha_t\}_{t=0}^{T-1}$ and $\{\beta_t\}_{t=0}^{T-1}$ be given sequences of positive numbers and*

$$\epsilon := \sum\nolimits_{t=0}^{T-1}\epsilon_t^2, \quad \alpha := \sum\nolimits_{t=0}^{T-1}\alpha_t, \quad \beta := \sum\nolimits_{t=0}^{T-1}\beta_t.$$

Suppose $\alpha+\beta<1$. If $\hat{u}_t$ is computed by (159) with n_t satisfying

$$n_t \ge \frac{\left(\hat{E}_t^r\sqrt{2\|\hat{\Omega}_t\|\log\frac{2m}{\beta_t}} + \left(\epsilon_t\hat{E}_t^r + \|\hat{E}_t^{ru}\|_\infty\right)\sqrt{\frac{1}{2}\log\frac{2}{\alpha_t}}\right)^2}{\epsilon_t^2(\hat{E}_t^r)^4} \tag{163}$$

and

$$\hat{E}_t^r > \sqrt{\frac{1}{2n_t}\log\frac{2}{\alpha_t}} \tag{164}$$

for all $t = 0,1,\ldots,T-1$, then

$$\|\hat{u}-u\|_\infty^2 := \sum_{t=0}^{T-1}\|\hat{u}_t - u_t\|_\infty^2 \le \epsilon \tag{165}$$

with probability greater than or equal to $1-\alpha-\beta$.

Proof. Using the definitions in (160), we have

$$\begin{aligned}\|u_t - \hat{u}_t\|_\infty =& \|E_t^{ru}/E_t^r - \hat{E}_t^{ru}/\hat{E}_t^r\|_\infty \\ =& \frac{1}{E_t^r\hat{E}_t^r}\left\|E_t^{ru}\hat{E}_t^r - \hat{E}_t^{ru}\hat{E}_t^r + \hat{E}_t^{ru}\hat{E}_t^r - \hat{E}_t^{ru}E_r\right\|_\infty \\ \le& \frac{1}{E_t^r\hat{E}_t^r}\left(\|E_t^{ru}\hat{E}_t^r - \hat{E}_t^{ru}\hat{E}_t^r\|_\infty + \|\hat{E}_t^{ru}\hat{E}_t^r - \hat{E}_t^{ru}E_r\|_\infty\right).\end{aligned} \tag{166}$$

The absolute value in the denominator is removed since $r(i) \in (0,1]$. Given α_t and n_t, choose $\gamma_1 = \sqrt{\frac{1}{2n_t}\log\frac{2}{\alpha_t}}$. Then (161) implies that

$$\left|\hat{E}_t^r - E_t^r\right| \le \sqrt{\frac{1}{2n_t}\log\frac{2}{\alpha_t}} \tag{167}$$

holds with probability at least $1-\alpha_t$. Combining with (164), we have

$$0 < \hat{E}_t^r - \sqrt{\frac{1}{2n_t}\log\frac{2}{\alpha_t}} \le E_t^r. \tag{168}$$

Similarly, choosing $\gamma_2 = \sqrt{\frac{2\|\hat{\Omega}_t\|}{n_t}\log\frac{2m}{\beta_t}}$, (162) implies that

$$\left\|\hat{E}_t^{ru} - E_t^{ru}\right\|_\infty \le \sqrt{\frac{2\|\hat{\Omega}_t\|}{n_t}\log\frac{2m}{\beta_t}} \tag{169}$$

holds with probability at least $1-\beta_t$. Applying (167), (168) and (169) to (166), and using union bound, we obtain that

$$\|u_t - \hat{u}_t\|_\infty \le \frac{\hat{E}_t^r\sqrt{\frac{2\|\hat{\Omega}_t\|}{n_t}\log\frac{2m}{\beta_t}} + \|\hat{E}_t^{ru}\|_\infty\sqrt{\frac{1}{2n_t}\log\frac{2}{\alpha_t}}}{\left(\hat{E}_t^r - \sqrt{\frac{1}{2n_t}\log\frac{2}{\alpha_t}}\right)\hat{E}_t^r} \tag{170}$$

with probability at least $1-\alpha_t-\beta_t$. Hence we have

$$\Pr\left(\|u_t-\hat{u}_t\|_\infty^2 \le \epsilon_t^2\right) \ge 1-\alpha_t-\beta_t \tag{171}$$

if the right hand side of (170) is less than or equal to ϵ_t. Solving this condition for n_t, we get (163). Therefore, we conclude that if n_t satisfies (163) and (164), we obtain (171). Using the union bound on (171), we get the desired results. □

Remark 8. *The sample complexity in (163) reveals that the required number of samples for path integral control depends only logarithmically on the dimension m.*

7.5.2 Upper Bound on the Control Performance

Often in practice, we are interested in an upper bound on the performance loss of the control system. To obtain such a bound, we now analyze the impact of $\|\hat{u}-u\|_\infty$ on the control performance.

Let $\hat{U}_t$ be the "noisy" control input determined by the path integral controller. The closed-loop dynamics is

$$X_{t+1} = A_tX_t + B_t\hat{U}_t + W_t, \quad W_t \sim \mathcal{N}(0,\Omega_t) \tag{172}$$

and the accrued LQR cost is

$$\mathcal{L} := \mathbb{E}\left[\sum_{t=0}^{T-1}\left(\frac{1}{2}X_t^\top M_tX_t + \frac{1}{2}\hat{U}_t^\top N_t\hat{U}_t\right) + \frac{1}{2}X_T^\top M_TX_T\right]. \tag{173}$$

Introducing $V_t := \hat{U}_t - U_t$, (172) can be rewritten as

$$X_{t+1} = \widetilde{A}_tX_t + B_tV_t + W_t, \quad W_t \sim \mathcal{N}(0,\Omega_t) \tag{174}$$

where we introduced $\widetilde{A}_t := A_t + B_tK_t$. Also, (173) can be written in terms of V_t as

$$\mathcal{L} = \mathbb{E}\sum_{t=0}^{T-1}\left(\frac{1}{2}X_t^\top\widetilde{M}_tX_t + X_t^\top\widetilde{N}_tV_t + \frac{1}{2}V_t^\top N_tV_t\right) + \mathbb{E}\left[\frac{1}{2}X_T^\top M_TX_T\right]$$

where $\widetilde{M}_t := M_t + K_t^\top N_tK_t$ and $\widetilde{N}_t := K_t^\top N_t$. From Theorem 16, we know that if n_t satisfies (163) and (164) then $\sum_{t=0}^{T-1}\|v_t\|_\infty^2 \le \epsilon$ with probability at least $1-\alpha-\beta$. Now we formulate the problem to search for the state feedback policy $v_t = \pi_t(x_t)$ that maximizes $\mathcal{L}$ while satisfying $\sum_{t=0}^{T-1}\|v_t\|_\infty^2 \le \epsilon$ with probability at least $1-\alpha-\beta$. The problem is formulated as the following chance-constrained optimization problem.

Problem 8 (Chance-constrained LQR)**.**

$$\begin{aligned}\max_{\{\pi_t(\cdot)\}_{t=0}^{T-1}} \quad & \mathbb{E}\sum_{t=0}^{T-1}\left(\frac{1}{2}X_t^\top\widetilde{M}_tX_t + X_t^\top\widetilde{N}_tV_t + \frac{1}{2}V_t^\top N_tV_t\right) + \mathbb{E}\left[\frac{1}{2}X_T^\top M_TX_T\right] \\ \text{s.t.} \quad & X_{t+1} = \widetilde{A}_tX_t + B_tV_t + W_t, \quad W_t \sim \mathcal{N}(0,\Omega_t) \\ & Pr\left(\sum_{t=0}^{T-1}\|v_t\|_\infty^2 \le \epsilon\right) \ge 1-\alpha-\beta.\end{aligned} \tag{175}$$

Let f^* be the value of the optimization problem 8. Then, if n_t satisfies (163) and (164), $\mathcal{L} \leq f^*$. Finding a worst-case policy π_t that solves the Problem 8 involving a chance constraint on the control input of the systems is inherently challenging. Some approaches have been proposed in the literature to solve the optimization problems with a chance constraint on the state of the system [41, 119]. Finding the solution to Problem 8 is left as a topic for future work.

7.6 Simulation Results

Consider an LQR problem where $A_t = \begin{bmatrix} 0.9 & -0.1 \\ -0.1 & 0.8 \end{bmatrix}$, $B_t = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\Omega_t = \begin{bmatrix} 4 & 0 \\ 0 & 0 \end{bmatrix}$, $M_t = 0.1I$, $N_t = 10$. I represents an identity matrix of size 2×2. In figure 25, we plot the state and control input trajectories for two values of n. The red dashed lines represent the analytical solution obtained by the classical Riccati equation. The blue solid lines represent the solution obtained by path integral. As evident from the figure, as n is increased, the trajectories obtained by path integral become less noisy and align well with the analytical LQR solution. We also plot in figures 25(b) and 25(d) the bounds $u_t \pm \epsilon_t$. As one can see the error ϵ_t reduces as n is increased. Figure 26(a) represents how ϵ_0 changes with the sample size. Figure 26(b) represents the LQR costs obtained by the classical Riccati equation (red dotted line) and by the path integral approach (blue solid line) as functions of the sample size. Notice that the path integral solution converges to the analytical solution as the sample size is increased.

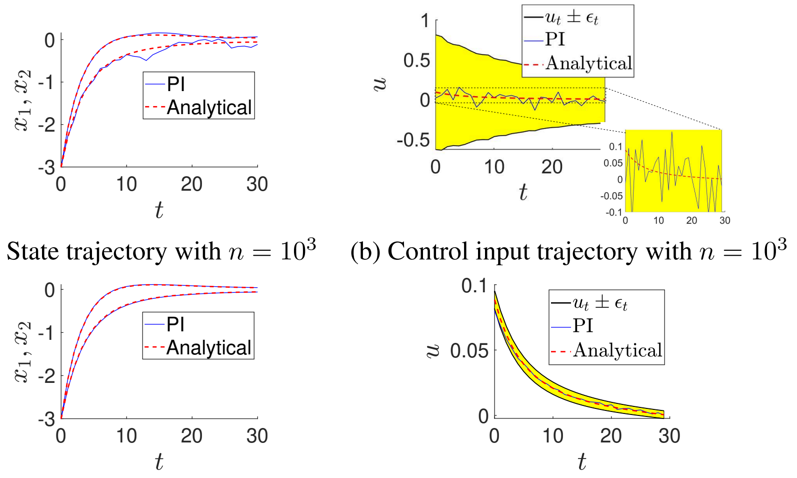


Figure 25: State and input trajectories for two values of n. The red dashed line represents the solution obtained by the Riccati equation, whereas the blue solid line represents the solution obtained by path integral control. The bounds $u_t \pm \epsilon_t$ are plotted in (b) and (d).

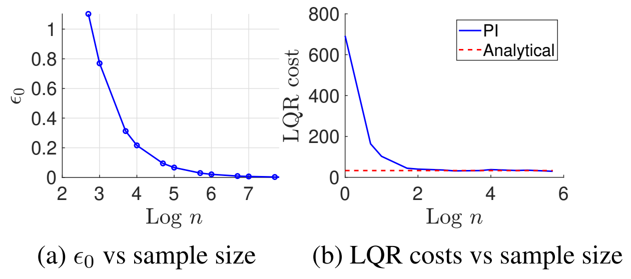


Figure 26: ϵ_0 and LQR cost vs sample size

7.7 Publications

- **A. Patil**, G. Hanasusanto, T. Tanaka, “Discrete-Time Stochastic LQR via Path Integral Control and Its Sample Complexity Analysis,” *IEEE Control Systems Letters (L-CSS)*
- **A. Patil**, G. Hanasusanto, T. Tanaka, “Discrete-Time Stochastic LQR via Path Integral Control and Its Sample Complexity Analysis,” *2024 IEEE Conference on Decision and Control (CDC)*

7.8 Future Work

Future work will build upon the foundation presented in this proposal to carry out sample complexity analysis of path integral for nonlinear continuous-time stochastic control problems. Another direction we would like to work on is to “robustify” the path integral control method by exploiting techniques from H^∞ control.

8 Proposed Dissertation Chapters and Timeline

This section introduces the dissertation chapters. Table. 1 shows a timeline of proposed tasks in red and writing in blue.

Chapter 1: Introduction

This chapter first describes the motivation of the dissertation and then introduces the basics of stochastic optimal control problems and path integral control.

Chapter 2: Continous-Time Chance-Constrained Stochastic Optimal Control

This chapter focuses on a continuous-time chance-constrained SOC problem, leveraging the path integral method to numerically solve the problem using open-loop samples of system trajectories. The proposed contributions include formulating the problem through stochastic calculus, proving strong duality between the primal and dual problems, and developing a novel dual ascent algorithm for real-time control, with simulation results in robot motion planning.

Chapter 3: Two-Player Zero-Sum Stochastic Differential Game

This chapter addresses two-player zero-sum stochastic differential games, where players make decisions in the presence of uncertainties, aiming to minimize their failure probabilities and control costs. A risk-minimizing approach is proposed, where the game is solved through a Hamilton-Jacobi-Isaacs partial differential equation with Dirichlet boundary conditions. The path integral control method is employed to solve this problem online, with no need for precomputation, and validated through examples like pursuit-evasion games and disturbance attenuation problems.

Chapter 4: Deceptive Control

This chapter addresses a deception problem between a supervisor and an agent, where the agent deviates from the supervisor’s reference policy to achieve its own goal while minimizing the supervisor’s detection. Using hypothesis testing and KL divergence minimization, the agent’s optimal deceptive policy is synthesized through path integral control, allowing for efficient online computation using Monte Carlo simulations.

Chapter 5: Task Hierarchical Control

Robotic systems with many degrees of freedom often require hierarchical control to manage multiple tasks with different priorities. This chapter integrates the null-space projection technique with the path integral control method, combining the computational efficiency of local controllers with the optimal control capabilities of the path integral approach, enabling real-time task prioritization in complex, multi-task robotic systems.

Dissertation	Dec'24	Jan'25	Feb'25	Mar'25	Apr'25	May'25
Chapter 1	Writing					
Chapter 2		Writing				
Chapter 3					Writing	
Chapter 4					Writing	
Chapter 5					Writing	
Chapter 6	Task 6.1	Task 6.2	Task 6.3	Task 6.3		Writing
Chapter 7						Writing

Table 1: 6-Month Timeline of Dissertation Completion

Chapter 6: Detection and Risk Mitigation of Stealthy Attack

The focus of this chapter will be on solving the "Detection and Risk Mitigation of Stealthy Attack" problem using the path integral approach, extending our prior work on deceptive control and two-player zero-sum games. The proposed framework will model the interaction between a system operator and an attacker, optimizing both attack detection and risk mitigation strategies by solving KL control problems. We also plan to conduct simulation studies using the simulator Isaac Sim and real-world experiments on a Qbot mobile robot and a quadruped robot, Spot.

Chapter 7: Sample Complexity Analysis of Path Integral Controller

This chapter introduces a theoretical framework for analyzing the sample complexity of the path integral control method, focusing on discrete-time stochastic LQR problems. The contributions include deriving a discrete-time path integral formulation, establishing an end-to-end error bound for control signals based on sample sizes, and formulating a chance-constrained optimization problem to assess the worst-case control performance, laying the groundwork for future analysis in nonlinear systems.

9 Publications

This section includes all the publications from my PhD studies.

9.1 Journal Publications

- **A. Patil**, G. Hanasusanto, T. Tanaka, "Discrete-Time Stochastic LQR via Path Integral Control and Its Sample Complexity Analysis," *IEEE Control Systems Letters (L-CSS)*
- **A. Patil**, A. Duarte, F. Bisetti, T. Tanaka, "Strong Duality and Dual Ascent Approach to Continuous-Time Chance-Constrained Stochastic Optimal Control," *submitted to Transactions on Automatic Control*
- M. Baglioni, **A. Patil**, L. Sentis, A. Jamshidnejad "Achieving Multi-UAV Best Viewpoint Coordination in Obstructed Environments," *under preparation*

9.2 Conference Publications

- **A. Patil**, G. Hanasusanto, T. Tanaka, "Discrete-Time Stochastic LQR via Path Integral Control and Its Sample Complexity Analysis," *2024 IEEE Conference on Decision and Control (CDC)*
- **A. Patil***, M. Karabag*, U. Topcu, T. Tanaka, "Simulation-Driven Deceptive Control via Path Integral Approach," *2023 IEEE Conference on Decision and Control (CDC)*
- **A. Patil**, Y. Zhou, D. Fridovich-Keil, T. Tanaka, "Risk-Minimizing Two-Player Zero-Sum Stochastic Differential Game via Path Integral Control," *2023 IEEE Conference on Decision and Control (CDC)*
- **A. Patil**, T. Tanaka, "Upper and Lower Bounds for End-to-End Risks in Stochastic Robot Navigation," *2023 IFAC World Congress*

- **A. Patil**, A. Duarte, A. Smith, F. Bisetti, T. Tanaka, "Chance-Constrained Stochastic Optimal Control via Path Integral and Finite Difference Methods," *2022 IEEE Conference on Decision and Control (CDC)*
- **A. Patil**, T. Tanaka, "Upper Bounds for Continuous-Time End-to-End Risks in Stochastic Robot Navigation," *2022 European Control Conference (ECC)*
- **A. Patil**, R. Funada, T. Tanaka, L. Sentis, "Task Hierarchical Control via Null-Space Projection and Path Integral Approach," *submitted to 2024 American Control Conference (ACC)*
- M. Baglioni, **A. Patil**, L. Sentis, A. Jamshidnejad "Achieving Multi-UAV Best Viewpoint Coordination in Obstructed Environments," *submitted to 2024 American Control Conference (ACC)*

10 Summary

The summary of this dissertation proposal is depicted in Figure 27. The proposal focuses on advancing path integral control theory for solving Stochastic Optimal Control (SOC) problems, which are prevalent in systems influenced by uncertainties and random disturbances, such as autonomous robotics. Traditional SOC methods like dynamic programming face computational challenges, particularly in high-dimensional, nonlinear systems. The path integral approach offers a promising alternative by transforming the SOC problem into an expectation over noisy system trajectories, allowing for efficient computation using Monte Carlo sampling. The dissertation explores and develops the path integral control theory for six different classes of SOC problems as shown in Figure 27, with an emphasis on real-time applications and scalability through parallel computing on GPUs. Through these contributions, the research enhances the utility of the path integral control method for handling complex, nonlinear, and safety-critical systems. This work also presents a sample complexity analysis and outlines future directions for developing autonomous systems based on path integral control theory.

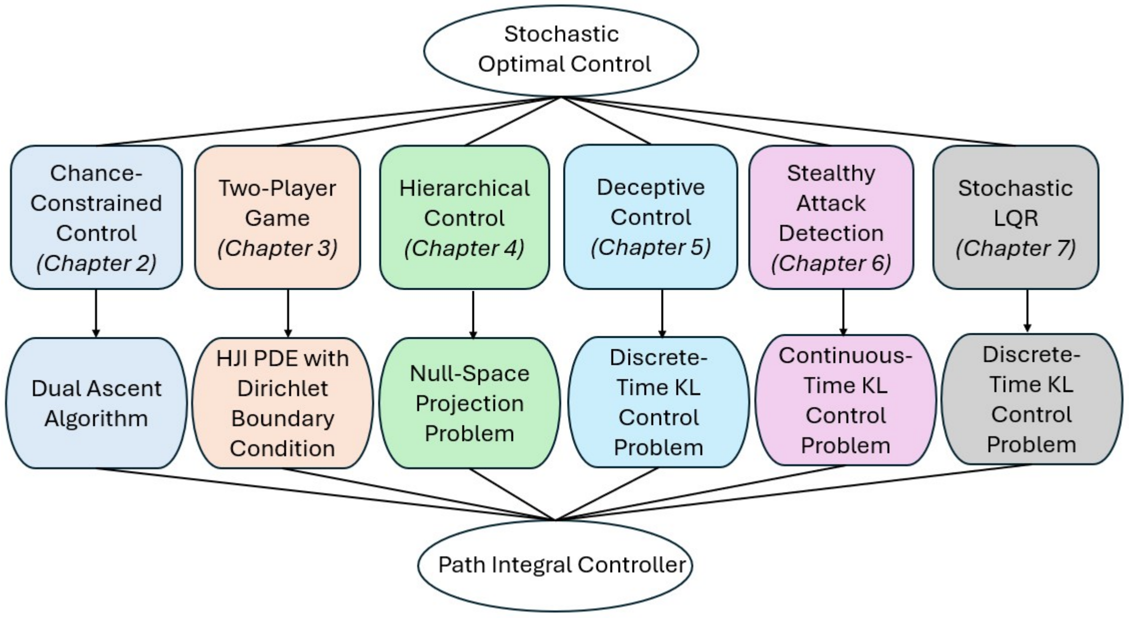

Stochastic Optimal Control
Chance-Constrained Control (Chapter 2)
Two-Player Game (Chapter 3)
Hierarchical Control (Chapter 4)
Deceptive Control (Chapter 5)
Stealthy Attack Detection (Chapter 6)
Stochastic LQR (Chapter 7)
Dual Ascent Algorithm
HJI PDE with Dirichlet Boundary Condition
Null-Space Projection Problem
Discrete-Time KL Control Problem
Continuous-Time KL Control Problem
Discrete-Time KL Control Problem
Path Integral Controller

Figure 27: Summary of the dissertation proposal

11 Appendix

11.1 Nonconvex Optimization and Strong Duality

Let $\mathcal{X}$ be a vector space. Suppose f_0 and f_1 are not necessarily convex, real-valued functions defined on $\mathcal{X}$. The domains of the functions f_0 and f_1 are denoted by $\text{dom}(f_0)$ and $\text{dom}(f_1)$, respectively. Consider the following optimization problem with a single inequality constraint:

$$\min_{x\in\mathcal{X}} \quad f_0(x) \tag{176a}$$
$$f_1(x) \leq 0. \tag{176b}$$

We assume $\mathrm{dom}(f_0)\cap\mathrm{dom}(f_1)$ is nonempty, and $f_0(x) > -\infty \ \ \forall x\in\mathrm{dom}(f_0)\cap\mathrm{dom}(f_1)$.

Let $\eta \geq 0$ be the dual variable. Define the dual function $g : [0,\infty) \to \bar{\mathbb{R}}$ by

$$g(\eta) = \inf_{x\in\mathcal{X}} \ f_0(x) + \eta f_1(x). \tag{177}$$

Since (177) is a pointwise infimum of a family of affine functions, it is concave. Moreover, since affine functions are upper semicontinuous, $g(\eta)$ is also upper semicontinuous.

Assumption 7. *Problem (176) is strictly feasible. That is, there exists $x_0 \in$ dom(f_0) such that $f_1(x_0) < 0$.*

Lemma 3. *Under Assumption 7, there exists a dual optimal solution η^* such that $0 \leq \eta^* < \infty$ such that $g(\eta^*) = \sup_{\eta\geq 0} g(\eta)$.*

Proof. Since $g(\eta)$ is upper semicontinuous, by Weierstrass' theorem [120, Proposition A.8], it is sufficient to prove that the function $-g(\eta)$ is coersive. Specifically, it is sufficient to show that there exists a scalar γ such that the set

$$\eta(\gamma) := \{\eta \geq 0 \mid g(\eta) \geq \gamma\}$$

is nonempty and compact.

We will show that the choice $\gamma := \inf_{x\in\mathcal{X}} f_0(x)$ will make $\eta(\gamma)$ nonempty and bounded. To see that $\eta(\gamma)$ is nonempty, notice that

$$g(0) = \inf_{x\in\mathcal{X}} f_0(x) = \gamma$$

and thus $0 \in \eta(\gamma)$.

To prove $\eta(\gamma)$ is bounded, we need to show that $g(\eta) \searrow -\infty$ as $\eta \to +\infty$. Let x_0 be a strictly feasible solution, i.e., $f_1(x_0) < 0$. We have

$$\begin{aligned} g(\eta) &= \inf_{x} \ f_0(x) + \eta f_1(x) \\ &\leq f_0(x_0) + \eta \underbrace{f_1(x_0)}_{<0} \to -\infty \end{aligned}$$

as $\eta \to +\infty$. This completes the proof. □

To proceed further, we need an additional set of assumptions.

Assumption 8. *(i) For each $\eta \geq 0$, the set*

$$X(\eta) := \operatorname*{argmin}_{x\in\mathcal{X}} \ f_0(x) + \eta f_1(x)$$

is nonempty.

(ii) The function $\eta \mapsto f_1(X(\eta)) : [0,\infty) \to \mathbb{R}$ is well-defined. That is, if $x_1 \in X(\eta)$ and $x_2 \in X(\eta)$, then $f_1(x_1) = f_2(x_2)$.

(iii) The function $f_1(X(\cdot))$ is continuous.

Lemma 4. *Suppose Assumption 7 holds so that a dual optimal solution $0 \leq \eta^* < +\infty$ exists (c.f., Lemma 3). The following statements hold:*

(i) If $\eta^* = 0$, then $f_1(X(\eta^*)) \leq 0$.

(ii) If $\eta^* > 0$, then $f_1(X(\eta^*)) = 0$.

Proof. (i) Suppose $\eta^* = 0$ but $f_1(X(\eta^*)) > 0$. Set $\eta_\epsilon = \epsilon > 0$. By continuity of $f_1(X(\cdot))$, we have $f_1(X(\eta_\epsilon)) > 0$ for a sufficiently small ϵ. Notice that

$$\begin{aligned} g(\eta_\epsilon) &= f_0(X(\eta_\epsilon)) + \underbrace{\eta_\epsilon f_1(X(\eta_\epsilon))}_{>0} \\ &> f_0(X(\eta_\epsilon)) \\ &\geq \min_{x\in\mathcal{X}}\ f_0(x) \\ &= \min_{x\in\mathcal{X}}\ f_0(x) + \eta^* f_1(x) \\ &= g(\eta^*). \end{aligned}$$

However, this chain of inequalities contradicts to the fact that η^* is the dual optimal solution (i.e., η^* maximizes $g(\eta)$).

(ii) Suppose $\eta^* > 0$ and $f_1(X(\eta^*)) > 0$. Set $\eta_\epsilon = \eta^* + \epsilon$ where $\epsilon > 0$ is sufficiently small. By continuity of $f_1(X(\cdot))$, we have $f_1(X(\eta_\epsilon)) > 0$. Notice that

$$\begin{aligned} g(\eta_\epsilon) &= f_0(X(\eta_\epsilon)) + \eta_\epsilon f_1(X(\eta_\epsilon)) \\ &= f_0(X(\eta_\epsilon)) + \eta^* f_1(X(\eta_\epsilon)) + \underbrace{\epsilon f_1(X(\eta_\epsilon))}_{>0} \\ &> f_0(X(\eta_\epsilon)) + \eta^* f_1(X(\eta_\epsilon)) \\ &\geq \min_{x\in\mathcal{X}}\ f_0(x) + \eta^* f_1(x) \\ &= g(\eta^*). \end{aligned}$$

This contradicts to the fact that η^* maximizes $g(\eta)$.

Similarly, suppose $\eta^* > 0$ and $f_1(X(\eta^*)) < 0$. Set $\eta_\epsilon = \eta^* - \epsilon$ and choose $\epsilon > 0$ sufficiently small so that $\eta_\epsilon > 0$ and $f_1(X(\eta_\epsilon)) < 0$. We have

$$\begin{aligned} g(\eta_\epsilon) &= f_0(X(\eta_\epsilon)) + \eta_\epsilon f_1(X(\eta_\epsilon)) \\ &= f_0(X(\eta_\epsilon)) + \eta^* f_1(X(\eta_\epsilon)) - \epsilon \underbrace{f_1(X(\eta_\epsilon))}_{<0} \\ &> f_0(X(\eta_\epsilon)) + \eta^* f_1(X(\eta_\epsilon)) \\ &\geq \min_{x\in\mathcal{X}}\ f_0(x) + \eta^* f_1(x) \\ &= g(\eta^*). \end{aligned}$$

Again, this contradicts to the fact that η^* maximizes $g(\eta)$. Therefore, if $\eta^* > 0$ then $f_1(X(\eta^*)) = 0$ must hold. □

The following is the main result of this section.

Theorem 17. *Consider problem* (176) *and suppose Assumptions* 7 *and* 8 *hold. Then, there exists a dual optimal solution* $0 \leq \eta^* < +\infty$ *that maximizes* $g(\eta)$*. Moreover, the set*

$$X(\eta^*) := \operatorname*{argmin}_{x\in\mathcal{X}}\ f_0(x) + \eta^* f_1(x)$$

is nonempty, and any element $x^* \in X(\eta^*)$ *is a primal optimal solution such that* $f_0(x^*) = g(\eta^*)$*. i.e., the duality gap is zero. Consequently, a minimizer of* $f_0(x) + \eta^* f_1(x)$ *is an optimal solution of* (176)*.*

Proof. By Lemma 4, any element $x^* \in X(\eta^*)$ satisfies $f_1(x^*) \leq 0$. Hence, x^* satisfies primal feasibility. It also follows from Lemma 4 that $\eta^* f_1(x^*) = 0$, i.e., the complementary slackness condition holds. Therefore,

$$\begin{aligned} f_0(x^*) &= f_0(x^*) + \eta^* f_1(x^*) \quad \text{(Complementary slackness)} \\ &= \min_{x\in\mathcal{X}} f_0(x) + \eta^* f(x) \quad (\text{since } x^* \in X(\eta^*)) \\ &= g(\eta^*). \end{aligned}$$

Hence, the strong duality holds. Therefore, x^* is a primal optimal solution. It follows from the identity above that a minimizer of $f_0(x) + \eta^* f_1(x)$ is an optimal solution for (176). □

11.2 Legendre Duality

Let P and Q be probability distributions on $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$, and $C : \mathcal{X} \to \mathbb{R}$ a given cost function. Define the internal energy $U(Q, C)$, free energy $F(P, C)$ and relative entropy (KL divergence) $D(Q\|P)$ as:

$$\begin{aligned} U(Q,C) &:= \int_{\mathcal{X}} C(x)Q(dx) \\ F(P,C) &:= -\lambda \log \int_{\mathcal{X}} \exp\left(-\frac{C(x)}{\lambda}\right) P(dx) \\ D(Q\|P) &:= \int_{\mathcal{X}} \log \frac{dQ}{dP}(x)Q(dx). \end{aligned}$$

Then the following duality relationship holds:

$$\begin{aligned} F(P,C) &= \inf_{Q}\{U(Q,C) + \lambda D(Q\|P)\} \\ -\lambda D(Q\|P) &= \inf_{C}\{U(Q,C) - F(P,C)\}. \end{aligned}$$

Also, the optimal probability distribution Q^* is given by

$$Q^*(B) = \frac{\int_B \exp(-C(x)/\lambda)P(dx)}{\int_{\mathcal{X}} \exp(-C(x)/\lambda)P(dx)}, \quad \forall B \in \mathcal{B}(\mathcal{X}).$$

See [121, 34] for further discussions.

11.3 The Likelihood Ratio $\frac{dQ^*}{dP}(x)$

Let $Q(x)$ be the probability distribution of the trajectories defined by system (1) under the given policy $u(\cdot)$ and $Q^*(x)$ be the probability distribution of the trajectories defined by system (1) under the optimal policy $u^*(\cdot)$ Let $P(x)$ be the probability distribution of the trajectories defined by the uncontrolled system (2). Suppose the likelihood ratio of observing a sample path x under the distributions Q and P is denoted by $\frac{dQ}{dP}(x)$ and the expectation under any distribution Q is denoted by $\mathbb{E}^Q[\cdot]$. Let $k(t)$ be a process defined by

$$k(t) := \Sigma^{-1}Gu(t). \tag{178}$$

Now, we state the following theorem:

Theorem 18 (The Girsanov theorem)**.** *The likelihood ratio $\frac{dQ}{dP}(x)$ of observing a sample path x under distributions Q and P is given by*

$$\frac{dQ}{dP}(x) = \exp\left(\int_{t_0}^{t_f} k(t)^\top dw(t) + \frac{1}{2}\int_{t_0}^{t_f} \|k(t)\|^2 dt\right) \tag{179}$$

where the process $k(t)$ is defined by (178).

Proof. Refer to [59, Chapter 8.6]. □

Theorem 19. *Suppose we generate an ensemble of N trajectories $\{x^{(i)}\}_{i=1}^N$ under the distribution P. Let $r^{(i)}$ be the path reward associated with the trajectory $x^{(i)}$ as given in* (34)*. Define $r := \sum_{i=1}^N r^{(i)}$. Then as $N \to \infty$,*

$$\frac{r^{(i)}}{r/N} \xrightarrow{a.s.} \frac{dQ^*}{dP}(x)$$

Proof. The likelihood ratio $\frac{dQ}{dP}(x)$ of observing a sample path x is given by (179). Using this result, we obtain

$$\mathbb{E}^Q \log\left(\frac{dQ}{dP}(x)\right) = \frac{1}{2}\int_{t_0}^{t_f} \mathbb{E}^Q\|k(t)\|^2 dt. \tag{180}$$

In (180), we used the property of Itô integral [59, Chapter 3]:

$$\mathbb{E}^Q\left[\int_{t_0}^{t_f} k(t)^\top dw(t)\right] = 0$$

Now, for a given value of η, we wanted to solve the following problem

$$\min_{u(\cdot)} \mathbb{E}^{Q}_{x_0,t_0}\left[\phi(x(t_f);\eta)+\int_{t_0}^{t_f}\left(\frac{1}{2}u^\top Ru + V\right)dt\right]. \tag{181}$$

According to Assumption 1, there exists a positive constant λ such that

$$R = \lambda G^\top \Sigma^{-\top}\Sigma^{-1}G. \tag{182}$$

Now, we get

$$\min_{u(\cdot)} \mathbb{E}^{Q}_{x_0,t_0}\left[\phi(x(t_f);\eta)+\int_{t_0}^{t_f}\left(\frac{1}{2}u^\top Ru + V\right)dt\right] \tag{183a}$$

$$=\min_{u(\cdot)} \mathbb{E}^{Q}_{x_0,t_0}\left[\phi(x(t_f);\eta)+\int_{t_0}^{t_f}\left(\frac{1}{2}u^\top\lambda G^\top \Sigma^{-\top}\Sigma^{-1}Gu+V\right)dt\right] \tag{183b}$$

$$=\min_{k(\cdot)} \mathbb{E}^{Q}_{x_0,t_0}\left[\phi(x(t_f);\eta)+\int_{t_0}^{t_f}\left(\frac{\lambda}{2}\|k(t)\|^2 + V\right)dt\right] \tag{183c}$$

$$=\min_{Q(x)} \int_{\mathcal{T}}\left(\phi(x(t_f);\eta) + \int_{t_0}^{t_f} V dt + \lambda \log\frac{dQ}{dP}(x)\right)Q(dx). \tag{183d}$$

Equation (183b) is obtained by plugging (182) into (183a). (183c) obtained by using (178) and (183d) by using (180). Thus, we converted the problem (181) into a KL control problem. Invoking the Legendre duality between the KL divergence and free energy (See Appendix 11.2) it can be shown that there exists a minimizer Q^* of (183d) which can be written as

$$Q^*(dx) = \frac{\exp\left(-\frac{1}{\lambda}\left(\phi(x(t_f);\eta)+\int_{t_0}^{t_f} V dt\right)\right)P(dx)}{\mathbb{E}^P\left[\exp\left(-\frac{1}{\lambda}\left(\phi(x(t_f);\eta)+\int_{t_0}^{t_f} V dt\right)\right)\right]} \tag{184}$$

Using (184), we can write the expression for the Radon-Nikodym derivative $\frac{dQ^*}{dP}(x)$ as

$$\frac{dQ^*}{dP}(x) = \frac{\exp\left(-\frac{1}{\lambda}\left(\phi(x(t_f);\eta)+\int_{t_0}^{t_f} V dt\right)\right)}{\mathbb{E}^P\left[\exp\left(-\frac{1}{\lambda}\left(\phi(x(t_f);\eta)+\int_{t_0}^{t_f} V dt\right)\right)\right]} \tag{185}$$

□

Using (185), we can conclude that given the ensemble of N trajectories $\{x^{(i)}\}_{i=1}^N$ sampled under distribution P, the likelihood ratio $\frac{dQ^*}{dP}$ of observing a sample path $x^{(i)}$ is given by $\frac{r^{(i)}}{r/N}$ where $r^{(i)}$ is defined by (34) and $r := \sum_{i=1}^N r^{(i)}$. Using the strong law of large numbers as $N\to\infty$, we get

$$\frac{r^{(i)}}{r/N} \xrightarrow{a.s.} \frac{dQ^*}{dP}(x).$$

References

[1] R. Durrett, *Probability: theory and examples*, vol. 49. Cambridge university press, 2019.

[2] H. J. Kappen, "Path integrals and symmetry breaking for optimal control theory," *Journal of statistical mechanics: theory and experiment*, vol. 2005, no. 11, p. P11011, 2005.

[3] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *The Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[4] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1433–1440, IEEE, 2016.

[5] B. Øksendal, *Stochastic differential equations*. Springer, 2003.

[6] G. Williams, A. Aldrich, and E. A. Theodorou, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 344–357, 2017.

[7] A. Patil, G. A. Hanasusanto, and T. Tanaka, "Discrete-time stochastic lqr via path integral control and its sample complexity analysis," *IEEE Control Systems Letters*, 2024.

[8] E. Nelson, "Derivation of the Schrödinger equation from newtonian mechanics," *Physical review*, vol. 150, no. 4, p. 1079, 1966.

[9] E. Nelson, *Dynamical theories of Brownian motion*, vol. 106. Princeton university press, 2020.

[10] H. Rosenbrock, "A variational principle for quantum mechanics," *Physics letters A*, vol. 110, no. 7-8, pp. 343–346, 1985.

[11] H. H. Rosenbrock, "A stochastic variational treatment of quantum mechanics," *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 450, no. 1939, pp. 417–437, 1995.

[12] F. Guerra and L. M. Morato, "Quantization of dynamical systems and stochastic control theory," *Physical review D*, vol. 27, no. 8, p. 1774, 1983.

[13] K. Yasue, "Stochastic calculus of variations," *Journal of Functional Analysis*, vol. 41, no. 3, pp. 327–340, 1981.

[14] T. Itami, "Optimization of nonlinear control systems based on the principle of superposition (in japanese)," *Transactions of the Society of Instrument and Control Engineers*, vol. 37, no. 3, pp. 193–202, 2001. DOI:10.9746/sicetr1965.37.193.

[15] T. Itami, "Nonlinear optimal control via Monte-Carlo evaluation of path integrals (in japanese)," *Transactions of the Institute of Systems, Control and Information Engineers*, vol. 16, no. 12, pp. 637–648, 2003. DOI: 10.5687/iscie.16.637.

[16] W. H. Fleming and H. M. Soner, *Controlled Markov processes and viscosity solutions*, vol. 25. Springer Science & Business Media, 2006.

[17] P. Whittle, "Risk-sensitive linear/quadratic/Gaussian control," *Advances in Applied Probability*, vol. 13, no. 4, pp. 764–777, 1981.

[18] D. Jacobson, "Optimal stochastic linear systems with exponential performance criteria and their relation to deterministic differential games," *IEEE Transactions on Automatic control*, vol. 18, no. 2, pp. 124–131, 1973.

[19] I. R. Petersen, M. R. James, and P. Dupuis, "Minimax optimal control of stochastic uncertain systems with relative entropy constraints," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 398–412, 2000.

[20] E. Todorov, "Linearly-solvable Markov decision problems," *Advances in Neural Information Processing Systems*, vol. 19, 2006.

[21] E. Todorov, "Efficient computation of optimal actions," *Proceedings of the national academy of sciences*, vol. 106, no. 28, pp. 11478–11483, 2009.

[22] P. Piray and N. D. Daw, "Linear reinforcement learning in planning, grid fields, and cognitive control," *Nature communications*, vol. 12, no. 1, p. 4942, 2021.

[23] L. C. Evans, *Partial differential equations*, vol. 19. American Mathematical Society, 2022.

[24] E. Theodorou, J. Buchli, and S. Schaal, "Reinforcement learning of motor skills in high dimensions: A path integral approach," in *2010 IEEE International Conference on Robotics and Automation*, pp. 2397–2403, IEEE, 2010.

[25] P. Pastor, M. Kalakrishnan, S. Chitta, E. Theodorou, and S. Schaal, "Skill learning and task outcome prediction for manipulation," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3828–3834, IEEE, 2011.

[26] N. Sugimoto and J. Morimoto, "Phase-dependent trajectory optimization for CPG-based biped walking using path integral reinforcement learning," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pp. 255–260, IEEE, 2011.

[27] E. Rombokas, E. Theodorou, M. Malhotra, E. Todorov, and Y. Matsuoka, "Tendon-driven control of biomechanical and robotic systems: A path integral reinforcement learning approach," in *2012 IEEE International Conference on Robotics and Automation*, pp. 208–214, IEEE, 2012.

[28] M. Okada and T. Taniguchi, "Acceleration of gradient-based path integral method for efficient optimal and inverse optimal control," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3013–3020, IEEE, 2018.

[29] I. Abraham, A. Handa, N. Ratliff, K. Lowrey, T. D. Murphey, and D. Fox, "Model-based generalization under parameter uncertainty using path integral control," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2864–2871, 2020.

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

[31] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic MPC for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1714–1721, IEEE, 2017.

[32] S. Satoh and H. J. Kappen, "Nonlinear stochastic optimal control with input saturation constraints based on path integrals," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 15, no. 8, pp. 1169–1175, 2020.

[33] J. Carius, R. Ranftl, F. Farshidian, and M. Hutter, "Constrained stochastic optimal control with learned importance sampling: A path integral approach," *The International Journal of Robotics Research*, vol. 41, no. 2, pp. 189–209, 2022.

[34] E. A. Theodorou and E. Todorov, "Relative entropy and free energy dualities: Connections to path integral and KL control," *The 51st IEEE Conference on Decision and Control (CDC)*, pp. 1466–1473, 2012.

[35] H. J. Kappen, V. Gómez, and M. Opper, "Optimal control as a graphical model inference problem," *Machine learning*, vol. 87, pp. 159–182, 2012.

[36] S. Levine, "Reinforcement learning and control as probabilistic inference: Tutorial and review," *arXiv preprint arXiv:1805.00909*, 2018.

[37] S. Satoh, H. J. Kappen, and M. Saeki, "An iterative method for nonlinear stochastic optimal control based on path integrals," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 262–276, 2016.

[38] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, no. 10, pp. 1361–1379, 1996.

[39] Y. Kuwata, A. Richards, and J. How, "Robust receding horizon control using generalized constraint tightening," in *2007 American Control Conference*, pp. 4482–4487, IEEE, 2007.

[40] G. C. Calafiore and M. C. Campi, "The scenario approach to robust control design," *IEEE Transactions on automatic control*, vol. 51, no. 5, pp. 742–753, 2006.

[41] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.

[42] K. Oguri, M. Ono, and J. W. McMahon, "Convex optimization over sequential linear feedback policies with continuous-time chance constraints," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 6325–6331, IEEE, 2019.

[43] M. Vidyasagar, "Randomized algorithms for robust controller synthesis using statistical learning theory," *Automatica*, vol. 37, no. 10, pp. 1515–1528, 2001.

[44] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram, “Chance-constrained dynamic programming with application to risk-aware robotic space exploration,” *Autonomous Robots*, vol. 39, no. 4, pp. 555–571, 2015.

[45] A. Wang, A. Jasour, and B. C. Williams, “Non-gaussian chance-constrained trajectory planning for autonomous vehicles under agent uncertainty,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6041–6048, 2020.

[46] O. de Groot, B. Brito, L. Ferranti, D. Gavrila, and J. Alonso-Mora, “Scenario-based trajectory optimization in uncertain dynamic environments,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5389–5396, 2021.

[47] B. Li, Y. Tan, A.-G. Wu, and G.-R. Duan, “A distributionally robust optimization based method for stochastic model predictive control,” *IEEE Transactions on Automatic Control*, vol. 67, no. 11, pp. 5762–5776, 2021.

[48] A. Jasour, W. Han, and B. Williams, “Convex risk bounded continuous-time trajectory planning in uncertain non-convex environments,” in *2021 Robotics: Science and Systems*, 2021.

[49] Y. K. Nakka and S.-J. Chung, “Trajectory optimization for chance-constrained nonlinear stochastic systems,” in *2019 IEEE 58th conference on decision and control (CDC)*, pp. 3811–3818, IEEE, 2019.

[50] P. Hokayem, D. Chatterjee, and J. Lygeros, “Chance-constrained LQG with bounded control policies,” in *52nd IEEE Conference on Decision and Control*, pp. 2471–2476, IEEE, 2013.

[51] A. Nemirovski and A. Shapiro, “Convex approximations of chance constrained programs,” *SIAM Journal on Optimization*, vol. 17, no. 4, pp. 969–996, 2007.

[52] J. A. Paulson and A. Mesbah, “An efficient method for stochastic optimal control with joint chance constraints for nonlinear systems,” *International Journal of Robust and Nonlinear Control*, vol. 29, no. 15, pp. 5017–5037, 2019.

[53] K. M. Frey, T. J. Steiner, and J. How, “Collision probabilities for continuous-time systems without sampling,” *Proceedings of Robotics: Science and Systems. Corvalis, Oregon, USA (July 2020)*, 2020.

[54] A. Patil and T. Tanaka, “Upper and lower bounds for end-to-end risks in stochastic robot navigation,” *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 5603–5608, 2023.

[55] A. Patil and T. Tanaka, “Upper bounds for continuous-time end-to-end risks in stochastic robot navigation,” in *2022 European Control Conference (ECC)*, pp. 2049–2055, IEEE, 2022.

[56] X. Huang, M. Feng, A. Jasour, G. Rosman, and B. Williams, “Risk conditioned neural motion planning,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9057–9063, IEEE, 2021.

[57] A. Chern, X. Wang, A. Iyer, and Y. Nakahira, “Safe control in the presence of stochastic uncertainties,” in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 6640–6645, IEEE, 2021.

[58] S. K. Shah, C. D. Pahlajani, and H. G. Tanner, “Probability of success in stochastic robot navigation with state feedback,” in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3911–3916, IEEE, 2011.

[59] B. Oksendal, *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

[60] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[61] A. Friedman, *Stochastic differential equations and applications, vol. 1*. Academic Press, 1975.

[62] I. Karatzas and S. Shreve, *Brownian motion and stochastic calculus*, vol. 113. Springer Science & Business Media, 2012.

[63] R. M. Blumenthal and R. K. Getoor, *Markov processes and potential theory*. Courier Corporation, 2007.

[64] C. Grossmann, H.-G. Roos, and M. Stynes, *Numerical treatment of partial differential equations*, vol. 154. Springer, 2007.

[65] L. F. Shampine and M. W. Reichelt, “The MATLAB ODE suite,” *SIAM journal on scientific computing*, vol. 18, no. 1, pp. 1–22, 1997.

[66] H.-J. Yoon, C. Tao, H. Kim, N. Hovakimyan, and P. Voulgaris, "Sampling complexity of path integral methods for trajectory optimization," in *2022 American Control Conference (ACC)*, pp. 3482–3487, IEEE, 2022.

[67] T. Başar and G. J. Olsder, *Dynamic noncooperative game theory*. SIAM, 1998.

[68] P. J. Nahin, "Chases and escapes," in *Chases and Escapes*, Princeton University Press, 2012.

[69] W. Sun and P. Tsiotras, "Pursuit evasion game of two players under an external flow field," in *2015 American Control Conference (ACC)*, pp. 5617–5622, 2015.

[70] M. Falcone, "Numerical methods for differential games based on partial differential equations," *International Game Theory Review*, vol. 8, no. 02, pp. 231–272, 2006.

[71] H. Huang, J. Ding, W. Zhang, and C. J. Tomlin, "Automation-assisted capture-the-flag: A differential game approach," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 3, pp. 1014–1028, 2014.

[72] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on automatic control*, vol. 50, no. 7, pp. 947–957, 2005.

[73] D. Vrabie and F. Lewis, "Adaptive dynamic programming for online solution of a zero-sum differential game," *Journal of Control Theory and Applications*, vol. 9, no. 3, pp. 353–360, 2011.

[74] M. Prajapat, K. Azizzadenesheli, A. Liniger, Y. Yue, and A. Anandkumar, "Competitive policy optimization," in *Uncertainty in Artificial Intelligence*, pp. 64–74, PMLR, 2021.

[75] M. Liu, Y. Wan, F. L. Lewis, and V. G. Lopez, "Adaptive optimal control for stochastic multiplayer differential games using on-policy and off-policy reinforcement learning," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 12, pp. 5522–5533, 2020.

[76] X. Lin, P. A. Beling, and R. Cogill, "Multiagent inverse reinforcement learning for two-person zero-sum games," *IEEE Transactions on Games*, vol. 10, no. 1, pp. 56–68, 2017.

[77] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath, "Coherent measures of risk," *Mathematical finance*, vol. 9, no. 3, pp. 203–228, 1999.

[78] A. Dixit, M. Ahmadi, and J. W. Burdick, "Risk-averse receding horizon motion planning," *arXiv preprint arXiv:2204.09596*, 2022.

[79] D. Vrushabh, P. Akshay, K. Sonam, S. Wagh, and N. M. Singh, "Robust path integral control on stochastic differential games," in *2020 28th Mediterranean Conference on Control and Automation (MED)*, pp. 665–670, IEEE, 2020.

[80] T. Başar and P. Bernhard, *H-infinity optimal control and related minimax design problems: a dynamic game approach*. Springer Science & Business Media, 2008.

[81] R. Isaacs, *Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization*. Courier Corporation, 1999.

[82] A. Patil, A. Duarte, A. Smith, F. Bisetti, and T. Tanaka, "Chance-constrained stochastic optimal control via path integral and finite difference methods," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 3598–3604, IEEE, 2022.

[83] G. Antonelli, F. Arrichiello, and S. Chiaverini, "The null-space-based behavioral control for autonomous robotic systems," *Intelligent Service Robotics*, vol. 1, pp. 27–39, 2008.

[84] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[85] S. B. Slotine and B. Siciliano, "A general framework for managing multiple tasks in highly redundant robotic systems," in *proceeding of 5th International Conference on Advanced Robotics*, vol. 2, pp. 1211–1216, 1991.

[86] L. Sentis, M. Mintz, A. Ayyagari, C. Battles, S. Ying, and O. Khatib, "Large scale multi-robot coordination under network and geographical constraints," in *2009 IEEE International Symposium on Industrial Electronics*, pp. 1046–1053, IEEE, 2009.

[87] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 04, pp. 505–518, 2005.

[88] A. Dietrich, C. Ott, and A. Albu-Schäffer, "An overview of null space projections for redundant, torque-controlled robots," *The International Journal of Robotics Research*, vol. 34, no. 11, pp. 1385–1400, 2015.

[89] A. Patil, M. O. Karabag, T. Tanaka, and U. Topcu, "Simulator-driven deceptive control via path integral approach," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, pp. 271–277, IEEE, 2023.

[90] A. Patil, Y. Zhou, D. Fridovich-Keil, and T. Tanaka, "Risk-minimizing two-player zero-sum stochastic differential game via path integral control," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, pp. 3095–3101, IEEE, 2023.

[91] M. Otte and E. Frazzoli, "RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[92] B. Siciliano, "Kinematic control of redundant robot manipulators: A tutorial," *Journal of intelligent and robotic systems*, vol. 3, pp. 201–212, 1990.

[93] G. Antonelli, "Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems," *IEEE Transactions on Robotics*, vol. 25, no. 5, pp. 985–994, 2009.

[94] G. Antonelli, G. Indiveri, and S. Chiaverini, "Prioritized closed-loop inverse kinematic algorithms for redundant robotic systems with velocity saturations," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5892–5897, IEEE, 2009.

[95] Y. T. Tsai and D. E. Orin, "A strictly convergent real-time solution for inverse kinematics of robot manipulators," *Journal of robotic systems*, vol. 4, no. 4, pp. 477–501, 1987.

[96] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999.

[97] J. Bretagnolle and C. Huber, "Estimation des densités: risque minimax," *Séminaire de probabilités de Strasbourg*, vol. 12, pp. 342–363, 1978.

[98] J. Shim and R. C. Arkin, "A taxonomy of robot deception and its benefits in HRI," in *2013 IEEE international conference on systems, man, and cybernetics*, pp. 2328–2335, IEEE, 2013.

[99] A. Dragan, R. Holladay, and S. Srinivasa, "Deceptive robot motion: synthesis, analysis and experiments," *Autonomous Robots*, vol. 39, pp. 331–345, 2015.

[100] M. O. Karabag, M. Ornik, and U. Topcu, "Deception in supervisory control," *IEEE Transactions on Automatic Control*, vol. 67, no. 2, pp. 738–753, 2021.

[101] M. O. Karabag, M. Ornik, and U. Topcu, "Exploiting partial observability for optimal deception," *IEEE Transactions on Automatic Control*, 2022.

[102] M. Lloyd, *The art of military deception*. Pen and Sword, 2003.

[103] C. Wang and Z. Lu, "Cyber deception: Overview and the road ahead," *IEEE Security & Privacy*, vol. 16, no. 2, pp. 80–85, 2018.

[104] C. Keroglou and C. N. Hadjicostis, "Probabilistic system opacity in discrete event systems," *Discrete Event Dynamic Systems*, vol. 28, pp. 289–314, 2018.

[105] E. Kung, S. Dey, and L. Shi, "The performance and limitations of epsilon-stealthy attacks on higher order systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 2, pp. 941–947, 2016.

[106] C.-Z. Bai, F. Pasqualetti, and V. Gupta, "Data-injection attacks in stochastic control systems: Detectability and performance tradeoffs," *Automatica*, vol. 82, pp. 251–260, 2017.

[107] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, pp. 1889–1897, PMLR, 2015.

[108] S. Filippi, O. Cappé, and A. Garivier, "Optimism in reinforcement learning and Kullback-Leibler divergence," in *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 115–122, IEEE, 2010.

[109] E. Todorov, "Linearly-solvable Markov decision problems," *Advances in neural information processing systems*, pp. 1369–1376, 2007.

[110] K. Ito and K. Kashima, "Kullback–Leibler control for discrete-time nonlinear systems on continuous spaces," *SICE Journal of Control, Measurement, and System Integration*, vol. 15, no. 2, pp. 119–129, 2022.

[111] D. Bertsekas, *Dynamic programming and optimal control: Volume I*, vol. 1. Athena scientific, 2012.

[112] Z. Guo, D. Shi, K. H. Johansson, and L. Shi, "Worst-case stealthy innovation-based linear attack on remote state estimation," *Automatica*, vol. 89, pp. 117–124, 2018.

[113] J. Shang, H. Yu, and T. Chen, "Worst-case stealthy innovation-based linear attacks on remote state estimation under Kullback–Leibler divergence," *IEEE Transactions on Automatic Control*, 2021.

[114] K. Ito and K. Kashima, "Kullback-leibler control for discrete-time nonlinear systems on continuous spaces," 2022.

[115] O. Arslan, E. A. Theodorou, and P. Tsiotras, "Information-theoretic stochastic optimal control via incremental sampling-based algorithms," in *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pp. 1–8, IEEE, 2014.

[116] D. Bertsekas and S. Shreve, *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific, 1996.

[117] W. Hoeffding, "Probability inequalities for sums of bounded random variables," in *The collected works of Wassily Hoeffding*, pp. 409–426, Springer, 1994.

[118] A. Prékopa, "Boole-Bonferroni inequalities and linear programming," *Operations Research*, vol. 36, no. 1, pp. 145–162, 1988.

[119] Z. Zhou and R. Cogill, "Reliable approximations of probability-constrained stochastic linear-quadratic control," *Automatica*, vol. 49, no. 8, pp. 2435–2439, 2013.

[120] D. P. Bertsekas, "Nonlinear programming," *Journal of the Operational Research Society*, vol. 48, no. 3, pp. 334–334, 1997.

[121] M. Boué and P. Dupuis, "A variational representation for certain functionals of Brownian motion," *The Annals of Probability*, vol. 26, no. 4, pp. 1641–1659, 1998.