Probabilistically-sound and Asymptotically-optimal Algorithm for Stochastic Control with Trajectory Constraints

Vu Anh Huynh Emilio Frazzoli

***Abstract*— In this paper, we consider a class of stochastic optimal control problems with trajectory constraints. As a special case, we can constrain the probability that a system enters undesirable regions to remain below a certain threshold. We extend the incremental Markov Decision Process (iMDP) algorithm, which is a new computationally-efficient and asymptotically-optimal sampling-based tool for stochastic optimal control, to approximate arbitrarily well an optimal feedback policy of the constrained problem. We show that with probability one, in the presence of trajectory constraints, the sequence of policies returned from the algorithm is both probabilistically sound and asymptotically optimal. We demonstrate the proposed algorithm on motion planning and control problems subject to bounded collision probability in uncertain cluttered environments.**

I. Introduction

Planning and controlling dynamical systems in uncertain environments is a fundamental and essential problem in several fields, ranging from autonomous urban navigation, robotics [1], [2] to management science, economics, finance [3], [4] and healthcare [5], [6]. Given a system with dynamics described by a controlled diffusion process, the stochastic control problem is to find an optimal feedback policy to optimize an objective function.

It is well known that closed-form or exact algorithmic solutions for general continuous-time, continuous-space stochastic optimal control problems are computationally challenging [7]. Thus, many approaches have been proposed to investigate approximate solutions. Deterministic approaches such as discrete Markov Decision Process approximation [8], [9] and solving the associated Hamilton-Jacobi-Bellman PDE [10]–[12]) have been proposed, but the complexities of these approaches scale poorly with the dimension of the state space. Remarkably, as noted in [7], [13], [14], randomized algorithms provides a possibility to alleviate the curse of dimensionality by sampling the state space while assuming discrete control inputs.

Sampling-based algorithms can also be traced back to research in deterministic motion planning, which has been conducted in parallel with the stochastic optimal control research [15]–[17]. The deterministic motion planning problem aims to find a sequence of inputs that drives a system with noise-free dynamics from its initial condition to a goal region, while avoiding collision with obstacles. Sampling-based algorithms such as the Rapidly-exploring Random Tree (RRT) [16] and its asymptotically-optimal version RRT* [17] have been shown to be very effective for computing solutions to deterministic path planning in robotics on several platforms [1], [18]. This class of algorithms computes open-loop plans in the obstacle-free space by constructing exploring trees that require exact point-to-point steering from an initial state to a goal region. As a result, these algorithms are not aware of inherent uncertainty in system dynamics even when the system constantly re-plans after being out of its open-loop plans due to the underlying process noise. Therefore, RRT-like algorithms are not suitable for the purpose of stochastic optimal control.

Very recently, a novel computationally-efficient sampling-based algorithm called the incremental Markov Decision Process (iMDP) algorithm has been proposed to provide asymptotically-optimal solutions to a broad class of challenging stochastic optimal control problems [19]. Unlike exploring trees in RRT-like algorithms, the iMDP algorithm uses a different structure to address the difficulty caused by process noise. In particular, a sequence of finite-state Markov Decision Processes (MDPs) are generated to consistently approximate the original continuous-time stochastic dynamics. The finite models serve as incrementally refined models of the original problem. Consequently, distributions of approximating trajectories and control processes returned from these finite models approximate arbitrarily well distributions of optimal trajectories and optimal control processes of the original problem.

Compared to other algorithms for stochastic optimal control [8], [10]–[14], [20], the iMDP algorithm is the first practical algorithm that handles continuous time, continuous space as well as continuous control space. The enabling technical ideas lie in novel methods to compute Bellman updates. Moreover, the algorithm guarantees convergence to globally optimal solutions while maintaining low time and space complexity in the following sense. When the optimization over the continuous control space in the Bellman equation can be solved exactly, the sequence of computed approximate cost-to-go for each finite-state MDP converges almost surely to the optimal cost-to-go of the original continuous problem. When the mentioned optimization is solved by sampling controls, the above convergence happens in probability.

Although the iMDP algorithm provides asymptotically-optimal solutions to a *single* objective function, in practice, we are often concerned with *several* aspects of control policies represented by multiple functions of the controlled trajectories. For instance, an autonomous car aims to reach a goal with minimum time and at the same time minimize the risk of collision with obstacles. An effective way to

The authors are with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, 77 Massachusetts Ave., Cambridge, MA 02139. {vuhuynh, frazzoli}@mit.edu

model this situation is to optimize one of the functions and set the remaining functions within some accepted ranges as additional constraints. A possible method for solving the mentioned constrained optimization is to use the Lagrangian approach [21]–[23]. This approach requires numerical procedures to compute Lagrange multipliers before obtaining a policy, which is often computationally demanding for high dimensional systems.

In this paper, we formulate the stochastic optimal control problem with additional trajectory constraints that are expressed in terms of expected functions of the controlled trajectories. We extend the iMDP algorithm to solve the constrained optimization incrementally in order to provide anytime solutions after a small number of iterations. When more computing time is allowed, the proposed algorithm refines the solution quality in an efficient manner.

We note that trajectory constraints considered in this paper encode a large set of performance measures for a given policy. As a special case, we can constrain the probability that a system driven by a policy enters undesirable regions such as obstacles to remain below a certain threshold from *any starting state* in the state space. In this context, the considered problem is related to chance-constrained optimization in some previous works such as [24]–[26]. Our work, however, differs from these works in several aspects as follows.

First, in [24], [25], the authors consider discrete-time stochastic models constrained by bounded collision probability from *a particular starting state* and construct deterministic approximation by sampling entire trajectories from that state. In contrast, we compute the probability of collision induced by anytime feedback policies by building locally consistent transition probabilities. The advantage of our method is that the proposed algorithm constructs not only the optimal cost-to-go function but also the collision probability function over the entire state space under an anytime policy. Thus, the policy is feasible if the collision probability function is uniformly bounded by the specified threshold. As a result, there are regions in the state space where the associated collision probabilities are very small compared to the threshold, and therefore, the algorithm is able to find more aggressive controls in these regions.

Second, while the work presented in [26] can be used to compute an upper bound of collision probability for a given control policy, we present here a computationally-efficient algorithm that finds an asymptotically-optimal feedback policy and at the same time respects the collision probability constraint in a suitable sense.

The main contribution of this paper is an algorithm that guarantees probabilistically-sound and asymptotically-optimal solutions to the stochastic optimal control problem in the presence of trajectory constraints. That is, all constraints are satisfied with probability one, and the objective function is minimized as the number of iterations approaches infinity. We also show stronger results than those presented in [19], which assert the almost-sure convergence of approximating control policies to a globally optimal policy of the continuous problem even when the Bellman update equation is solved by sampling controls. We demonstrate the effectiveness of the proposed algorithm on motion planning and control problems subject to bounded collision probability in uncertain cluttered environments.

This paper is organized as follows. A formal problem definition is given in Section II. The extended iMDP algorithm is described in Section III. The analysis of the proposed algorithm is presented in Section IV. We present simulation examples and experimental results in Section V and conclude the paper in Section VI.

II. Problem Definition

In this section, we present a generic stochastic optimal control formulation with definitions and technical assumptions as discussed in [19], [27]. We also explain how to formulate trajectory constraints.

Stochastic Dynamics: Let d_x, d_u, and d_w be positive integers. Let S be a compact subset of $\mathbb{R}^{d_x}$, which is the closure of its interior S^o and has a smooth boundary ∂S. Let a compact subset U of $\mathbb{R}^{d_u}$ be a control set. The state of the system at time t is $x(t) \in S$, which is fully observable at all times.

Suppose that a stochastic process $\{w(t); t \geq 0\}$ is a d_w-dimensional Brownian motion on some probability space. Let a control process $\{u(t); t \geq 0\}$ be a U-valued, measurable process also defined on the same probability space such that the pair $(u(\cdot), w(\cdot))$ is admissible [19]. Let $\mathbb{R}^{d_x \times d_w}$ denote the set of all d_x by d_w real matrices. We consider systems with dynamics described by the controlled diffusion process:

$$dx(t) = f(x(t), u(t))\, dt + F(x(t), u(t))\, dw(t), \forall t \geq 0 \quad (1)$$

where $f : S \times U \to \mathbb{R}^{d_x}$ and $F : S \times U \to \mathbb{R}^{d_x \times d_w}$ are bounded measurable and continuous functions as long as $x(t) \in S^o$. The initial state $x(0)$ is a random vector in S. We assume that the matrix $F(\cdot, \cdot)$ has full rank. The continuity requirement of f and F can be relaxed with mild assumptions [19], [28].

We are interested in weak sense existence and weak sense uniqueness of solutions to Eq. 1, which assert the existence and uniqueness of the stochastic process $x(\cdot)$ via the existence and uniqueness of its probability distribution. As discussed in [19], [28], due to the boundedness of the set S, and the definition of the functions f and F in Eq. 1, we have a weak solution to Eq. 1 that is unique in the weak sense [29].

Policy, Cost-to-go Function and Trajectory Constraints: Markov controls are controls that depend only on the current state, i.e., $u(t)$ is a function only of $x(t)$, for all $t \geq 0$. A function $\mu : S \to U$ represents a Markov policy, which is known to be admissible with respect to the process noise $w(\cdot)$. Let Π be the set of all such policies. We define the *first exit time* $T_\mu : \Pi \to [0, +\infty]$ under policy μ as

$$T_\mu = \inf \big\{ t : x(t) \notin S^o \text{ and Eq. 1 and } u(t) = \mu(x(t)) \big\}.$$

In other words, T_μ is the first time that the trajectory of the dynamical system given by Eq. 1 with $u(t) = \mu(x(t))$ hits the boundary ∂S of S. The random variable T_μ can take value ∞ if the trajectory $x(\cdot)$ nevers exit S^o.

The expected cost-to-go function under a policy μ is a mapping from S to $\mathbb{R}$ defined as

$$J_\mu(z) = \mathbb{E}^z \left[\int_0^{T_\mu} \alpha^t \, g\big(x(t), \mu(x(t))\big) \, dt + h(x(T_\mu)) \right],$$

where $\mathbb{E}^z$ denotes the conditional expectation given $x(0) = z$, $g : S \times U \to \mathbb{R}$ and $h : S \to \mathbb{R}$ are bounded measurable and continuous functions, called the cost rate function and the terminal cost function, respectively, and $\alpha \in [0, 1)$ is the discount rate. We further assume that $g(x, u)$ is uniformly Hölder continuous in x with exponent $2\rho \in (0, 1]$ for all $u \in U$. That is, there exists some constant $\mathcal{K} > 0$ such that

$$|g(x, u) - g(x', u)| \leq \mathcal{K}||x - x'||_2^{2\rho}, \quad \forall x, x' \in S.$$

We consider additional trajectory constraints under a policy μ of the form $C_\mu(z) \in \Gamma$ *for all* z in S where

$$C_\mu(z) = \mathbb{E}^z \left[\int_0^{T_\mu} \beta^t \, r\big(x(t), \mu(x(t))\big) \, dt + k(x(T_\mu)) \right],$$

and $\Gamma \subset \mathbb{R}$ is some pre-specified accepted range. In the above definition, $r : S \times U \to \mathbb{R}$ and $k : S \times U \to \mathbb{R}$ are bounded measurable, continuous functions, and the discount rate β is also in $[0, 1)$. We note that the discontinuity of g, h, r and k can be treated as in [19], [28]. For simplicity, we consider one trajectory constraint in this paper, and handling multiple trajectory constraints is exactly the same.

Intuitively, the above constraint enforces the expected value, which is evaluated based on criteria encoded by $r(\cdot, \cdot)$ and $k(\cdot)$, over the distribution of controlled trajectories under the policy μ to be within Γ. Thus, the above formulation can be used to specify a broad set of constraints. For instance, in the context of autonomous cars, due to process noise, there may be a positive probability of collision for a given policy. When $\Gamma = (0, 0.001]$, $r(x, u) = 0$ for all x and u, and $k(x) = 1$ for $x \in \mathcal{X}_{obs}$ and 0 otherwise where $\mathcal{X}_{obs} \subset \partial S$ is the obstacle region, for large discount rate $\beta \in [0, 1)$, the trajectory constraint specifies (arbitrarily well) the probability that the trajectory $x(\cdot)$, starting from any state, collides with obstacles is less than or equal to 0.1 percent.

The *optimal cost-to-go function* $J^* : S \to \mathbb{R}$ is defined for all $z \in S$ as follows:

$$J^*(z) = \inf_{\mu \in \Pi} J_\mu(z) \text{ subject to } C_\mu(z) \in \Gamma \text{ and Eq. 1.}$$

A policy μ^* is called optimal if $J_{\mu^*} = J^*$. For any $\epsilon > 0$, a policy μ is called an ϵ-optimal policy if $||J_\mu - J^*||_\infty \leq \epsilon$.

We call a sampling-based algorithm probabilistically-sound if the probability that the solution returned by the algorithm is feasible approaches one as the number of samples increases. In addition, we call a sampling-based algorithm asymptotically-optimal if the sequence of solutions returned from the algorithm converges to an optimal solution in a suitable sense as the number of samples approaches infinity. Solutions returned from algorithms with the above properties are thus called probabilistically sound and asymptotically optimal.

In this paper, we consider the problem of computing the optimal cost-to-go function J^* and an optimal policy μ^* if obtainable. Our approach, outlined in Section III, approximates the optimal cost-to-go function and an optimal policy in an anytime fashion using an incremental sampling-based algorithm that is both probabilistically-sound and asymptotically-optimal.

III. Algorithm

In this section, we first overview the Markov chain approximation technique and then present the extended iMDP algorithm.

A. Markov Chain Approximation

A discrete-state Markov decision process (MDP) is a tuple $\mathcal{M} = (X, A, P, G, H)$ where X is a finite set of states, A is a set of actions that is possibly a continuous space, $P(\cdot \,|\, \cdot, \cdot) : X \times X \times A \to \mathbb{R}_{\geq 0}$ is the transition probability function, $G(\cdot, \cdot) : X \times A \to \mathbb{R}$ is an immediate cost function, and $H : X \to \mathbb{R}$ is a terminal cost function. From an initial state ξ_0, under a sequence of controls $\{v_i; i \in \mathbb{N}\}$, the induced trajectory $\{\xi_i; i \in \mathbb{N}\}$ is generated by following the transition probability function P.

The Markov chain approximation method approximates the continuous dynamics in Eq. 1 using a sequence of MDPs $\{\mathcal{M}_n = (S_n, U, P_n, G_n, H_n)\}_{n=0}^\infty$ and a sequence of holding times $\{\Delta t_n\}_{n=0}^\infty$ that are locally consistent. In particular, we construct $G_n(z, v) = g(z, v)\Delta t_n(z)$, $H_n(z) = h(z)$ for each $z \in S_n$ and $v \in U$. We also require that $\lim_{n\to\infty} \sup_{i \in \mathbb{N}, \omega \in \Omega_n} ||\Delta \xi_i^n||_2 = 0$ where Ω_n is the sample space of $\mathcal{M}_n$, $\Delta \xi_i^n = \xi_{i+1}^n - \xi_i^n$, and

- For all $z \in S$, $\lim_{n\to\infty} \Delta t_n(z) = 0$,
- For all $z \in S$ and all $v \in U$:

$$\lim_{n\to\infty} \frac{\mathbb{E}_{P_n}[\Delta \xi_i^n \,|\, \xi_i^n = z, u_i^n = v]}{\Delta t_n(z)} = f(z, v),$$

$$\lim_{n\to\infty} \frac{\mathrm{Cov}_{P_n}[\Delta \xi_i^n \,|\, \xi_i^n = z, u_i^n = v]}{\Delta t_n(z)} = F(z, v)F(z, v)^T.$$

The main idea of the Markov chain approximation approach for solving the original continuous problem is to solve a sequence of control problems defined on $\{\mathcal{M}_n\}_{n=0}^\infty$ as follows. A policy μ_n is a function that maps each state $z \in S_n$ to a control $\mu_n(z) \in U$. The set of all such policies is Π_n. We define $t_i^n = \sum_0^{i-1} \Delta t_n(\xi_i^n)$ for $i \geq 1$ and $t_0^n = 0$. Given a policy μ_n, the (discounted) cost-to-go due to μ_n is:

$$J_{n,\mu_n}(z) = \mathbb{E}_{P_n}^z \left[\sum_{i=0}^{I_n - 1} \alpha^{t_i^n} G_n(\xi_i^n, \mu_n(\xi_i^n)) + \alpha^{t_{I_n}^n} H_n(\xi_{I_n}^n) \right],$$

where $\mathbb{E}_{P_n}^z$ denotes the conditional expectation given $\xi_0^n = z$ under P_n, and $\{\xi_i^n; i \in \mathbb{N}\}$ is the sequence of states of the controlled Markov chain under the policy μ_n and I_n is termination time defined as $I_n = \min\{i : \xi_i^n \in \partial S_n\}$ where $\partial S_n = \partial S \cap S_n$. The continuous trajectory constraint is similarly approximated as $C_{n,\mu_n}(z) \in \Gamma$ where

$$C_{n,\mu_n}(z) = \mathbb{E}_{P_n}^z \left[\sum_{i=0}^{I_n - 1} \beta^{t_i^n} R_n(\xi_i^n, \mu_n(\xi_i^n)) + \beta^{t_{I_n}^n} K_n(\xi_{I_n}^n) \right],$$

where $R_n(x,v) = r(z,v)\Delta t_n(z)$, $K_n(z) = k(z)$ for $z \in S_n$ and $v \in U$.

The *optimal cost function*, denoted by J_n^*, satisfies

$$J_n^*(z) = \inf_{\mu_n \in \Pi_n} J_{n,\mu_n}(z) \text{ subject to } C_{n,\mu_n}(z) \in \Gamma,\ \forall z \in S_n.$$

An *optimal policy*, denoted by μ_n^*, satisfies $J_{n,\mu_n^*}(z) = J_n^*(z)$ for all $z \in S_n$. For any $\epsilon > 0$, μ_n is an ϵ-optimal policy if $||J_{n,\mu_n} - J_n^*||_\infty \leq \epsilon$.

The extension of iMDP outlined below is designed to compute the sequence of optimal cost-to-go $\{J_n^*\}_{n=0}^\infty$, the sequence of anytime control policies $\{\mu_n\}_{n=0}^\infty$ as well as the induced trajectory-constraint values $\{C_{n,\mu_n}(z)\}_{n=0}^\infty$ in an efficient iterative procedure.

B. Extension of iMDP

Before presenting the details of the algorithm, we discuss a number of primitive procedures. More details about these procedures can be found in [19], [27].

1) Sampling: The `Sample()` and `SampleBoundary()` procedures sample states independently and uniformly from the interior S^o and the boundary ∂S, respectively.

2) Nearest Neighbors: Given $z \in S$ and a set $Y \subseteq S$ of states. For any $k \in \mathbb{N}$, the procedure $\texttt{Nearest}(z, Y, k)$ returns the k nearest states $z' \in Y$ that are closest to z in terms of the Euclidean norm.

3) Time Intervals: Given a state $z \in S$ and a number $k \in \mathbb{N}$, the procedure $\texttt{ComputeHoldingTime}(z,k)$ returns a holding time computed as follows: $\texttt{ComputeHoldingTime}(z,k) = \gamma_t \left(\frac{\log k}{k}\right)^{\theta\varsigma\rho/d_x}$, where $\gamma_t > 0$ is a constant, and ς, θ are constants in $(0,1)$ and $(0,1]$ respectively. The parameter $\rho \in (0, 0.5]$ defines the Hölder continuity of the cost rate function $g(\cdot,\cdot)$ as in Section II.

4) Transition Probabilities: Given a state $z \in S$, a subset $Y \in S$, a control $v \in U$, and a positive number τ describing a holding time, the procedure $\texttt{ComputeTranProb}(z, v, \tau, Y)$ returns (i) a finite set $Z_{\text{near}} \subset S$ of states such that the state $z + f(z,v)\tau$ belongs to the convex hull of Z_{near} and $||z' - z||_2 = O(\tau)$ for all $z' \neq z \in Z_{\text{near}}$, and (ii) a function p that maps Z_{near} to a non-negative real numbers such that $p(\cdot)$ is a probability distribution over the support Z_{near}. It is crucial to ensure that these transition probabilities result in a sequence of locally consistent chains in the algorithm.

There are several ways to construct such transition probabilities. One possible construction by solving a system of linear equations can be found in [28] and is presented in [19], [27]. We can also compute the transition probabilities using local Gaussian distributions. We choose $Z_{\text{near}} = \texttt{Nearest}(z + f(z,v)\tau, Y, s)$ where $s = \Theta(\log(|Y|))$. Let $\mathcal{N}_{\overline{m},\sigma}(\cdot)$ denote the density of the (possibly multivariate) Gaussian distribution with mean $\overline{m}$ and variance σ. Define the transition probabilities as follows: $p(z') = \frac{\mathcal{N}_{\overline{m},\sigma}(z')}{\sum_{y \in Z_{\text{near}}} \mathcal{N}_{\overline{m},\sigma}(y)}$, where $\overline{m} = z + f(z,v)\tau$ and $\sigma = F(z,v)F(z,v)^T\tau$. This expression can be evaluated easily for any fixed $v \in U$. As $|Z_{near}|$ approaches infinity, the above construction satisfies the local consistency almost surely.

Algorithm 1: iMDP()

```
(n, S_0, J_0, μ_0, Δt_0) ← (1, ∅, ∅, ∅, ∅);
while n < N do
    (S_n, J_n, C_n, μ_n, Δt_n) ←
    (S_{n-1}, J_{n-1}, C_{n-1}, μ_{n-1}, Δt_{n-1});
    // Add a new state to the boundary
    z_s ← SampleBoundary();
    (S_n, J_n(z_s), C_n(z_s), μ_n(z_s), Δt_n(z_s)) ←
    (S_n ∪ {z_s}, h(z_s), k(z_s), ∅, 0) ;
    // Add a new state to the interior
    z_s ← Sample();
    z_nearest ← Nearest(z_s, S_n, 1);
    if (x_new, u_new, τ) ← ExtendBackwards(z_nearest, z_s, T_0) then
        z_new ← x_new(0);
        cost = τ g(z_new, u_new) + α^τ J_n(z_nearest);
        consValue = τ r(z_new, u_new) + β^τ C_n(z_nearest);
        // Discard if constraint value not in Γ
        if consValue ∉ Γ then
            continue ;
        (S_n, J_n(z_new), C_n(z_new), μ_n(z_new), Δt_n(z_new)) ←
        (S_n ∪ {z_new}, cost, consValue, u_new, τ) ;
        // Perform L_n ≥ 1 updates
        for i = 1 → L_n do
            // Choose K_n = Θ(|S_n|^θ) < |S_n| states
            Z_update ← Nearest(z_new, S_n \ ∂S_n, K_n) ∪ {z_new};
            for z ∈ Z_update do
                Update(z, S_n, J_n, μ_n, Δt_n);
    n ← n + 1;
```

5) Backward Extension: Given $T > 0$ and two states $z, z' \in S$, the procedure $\texttt{ExtendBackwards}(z, z', T)$ returns a triple (x, v, τ) such that (i) $\dot{x}(t) = f(x(t), u(t))dt$ and $u(t) = v \in U$ for all $t \in [0, \tau]$, (ii) $\tau \leq T$, (iii) $x(t) \in S$ for all $t \in [0, \tau]$, (iv) $x(\tau) = z$, and (v) $x(0)$ is close to z'. If no such trajectory exists, then the procedure returns failure. We can solve for the triple (x, v, τ) by sampling several controls v and choose the control resulting in $x(0)$ that is closest to z'.

6) Sampling and Discovering Controls: The procedure $\texttt{ConstructControls}(k, z, Y, T)$ returns a set of k controls in U. We can uniformly sample k controls in U. Alternatively, for each state $z' \in \texttt{Nearest}(z, Y, k)$, we solve for a control $v \in U$ such that (i) $\dot{x}(t) = f(x(t), u(t))dt$ and $u(t) = v \in U$ for all $t \in [0, T]$, (ii) $x(t) \in S$ for all $t \in [0, T]$, (iii) $x(0) = z$ and $x(T) = z'$.

The iMDP algorithm is presented in Algorithms 1-3. The algorithm incrementally refines a sequence of finite-state MDPs $\mathcal{M}_n = (S_n, U, P_n, G_n, H_n)$ and the associated holding time function Δt_n that consistently approximates the system in Eq. 1. In particular, given a state $z \in S_n$ and a holding time $\Delta t_n(z)$, we define the stage-cost function $G_n(z,v) = \Delta t_n(z) g(z,v)$ for all $v \in U$ and terminal-cost function $H_n(z) = h(z)$. Similarly, we define the trajectory-constraint stage-cost $R_n(z,v) = \Delta t_n(z) r(z,v)$, and trajectory-constraint terminal-cost $K_n(z) = k(z)$. We also associate with $z \in S_n$ a cost value $J_n(z)$, a control $\mu_n(z)$, and trajectory-constraint value $C_n(z)$. The functions J_n and C_n are referred to as cost value function and

Algorithm 2: `Update`$(z \in S_n, S_n, J_n, \mu_n, \Delta t_n)$

1 $\tau \leftarrow$ `ComputeHoldingTime`$(z, |S_n|)$;
`// Sample or discover` $M_n = \Theta(\log(|S_n|))$ `controls`
2 $U_n \leftarrow$ `ConstructControls`(M_n, z, S_n, τ);
3 **for** $v \in U_n$ **do**
4 $(Z_{\text{near}}, p_n) \leftarrow$ `ComputeTranProb`(z, v, τ, S_n);
5 $J \leftarrow \tau g(z, v) + \alpha^\tau \sum_{y \in Z_{\text{near}}} p_n(y) J_n(y)$;
6 $C \leftarrow \tau r(z, v) + \beta^\tau \sum_{y \in Z_{\text{near}}} p_n(y) C_n(y)$;
`// Improved cost and feasible constraint`
7 **if** $J < J_n(z)$ ***and*** $C \in \Gamma$ **then**
8 $(J_n(z), C_n(z), \mu_n(z), \Delta t_n(z)) \leftarrow (J, C, v, \tau)$;

Algorithm 3: `Policy`$(z \in S, n)$

1 $z_{\text{nearest}} \leftarrow$ `Nearest`$(z, S_n, 1)$;
2 **return** $\mu(z) = (\mu_n(z_{\text{nearest}}), \Delta t_n(z_{\text{nearest}}))$

constraint value function over S_n respectively.

Initially, an empty MDP model is created. In every main iteration of Algorithm 1, we construct a finer model based on the previous model. In particular, a state is sampled from the boundary of the state space (Lines 4-5). Subsequently, another state, z_{s}, is sampled from the interior of the state space S (Line 6). The nearest state z_{nearest} to z_{s} (Line 7) in the previous model is used to construct a new state z_{new} by using the procedure `ExtendBackwards` at Line 8. Unlike the original version of iMDP [19], we only accept z_{new} if an estimate of the associated constraint value belongs to the feasible set Γ (Line 13). This modification enables the sampling process to focus more on the state space region from which trajectories are likely to be feasible. Accepted new states are added to the state set, and their associated cost value $J_n(z_{\text{new}})$, constraint value $C_n(z_{\text{new}})$, and control $\mu_n(z_{\text{new}})$ are initialized at Line 14.

We then perform $L_n \geq 1$ updating rounds in each iteration (Lines 16-18). In particular, we construct the update-set Z_{update} consisting of $K_n = \Theta(|S_n|^\theta)$ states and z_{new} where $|K_n| < |S_n|$. For each of state z in Z_{update}, the procedure `Update` as shown in Algorithm 2 implements the following Bellman update:

$$J_n(z) = \min_{v \in \overline{U}(z)} \{G_n(z, v) + \alpha^{\Delta t_n(z)} \mathbb{E}_{P_n}[J_{n-1}(y)|z, v]\},$$

where

$$\overline{U}(z) = \{v \in U \,|R_n(z, v) + \beta^{\Delta t_n(z)} \mathbb{E}_{P_n}[C_{n-1}(y)|z, v] \in \Gamma\}.$$

The details of the implementation is as follows. A set of U_n controls is constructed using the procedure `ConstructControls` where $|U_n| = \Theta(\log(|S_n|))$ at Line 2. For each $v \in U_n$, we construct the support Z_{near} and compute the transition probability $P_n(\cdot \,|\, z, v)$ consistently over Z_{near} from the procedure `ComputeTranProb` (Line 4). The induced constraint values and cost values for the state z and controls in U_n are computed at Lines 6-5. We finally choose the best control in U_n that yields the smallest updated cost value and feasible constraint value (Line 8). As the current control may be still the best control compared to other controls in U_n, in Algorithm 2, we can re-evaluate the cost value and the constraint value with the current control $\mu_n(z)$ over the holding time $\Delta t_n(z)$ by adding the current control $\mu_n(z)$ to U_n. Essentially, we perform asynchronous policy evaluation to compute the constraint value function and perform asynchronous value iteration to compute the cost value function.

C. Feedback Control

We can perform a Bellman update based on the approximated cost-to-go J_n (using the stochastic continuous-time dynamics) to obtain a policy control for any n. However, we will discuss in Theorem 2 that the sequence of μ_n also approximates arbitrarily well an optimal control policy. In the following paragraph, we present an algorithm that converts a policy for a discrete system to a policy for the original continuous problem.

For each $n \in \mathbb{N}$, the control policy μ_n generated by the iMDP algorithm is used for controlling the original system described by Eq. 1 using the procedure described in Algorithm 3. This procedure computes the state in $\mathcal{M}_n$ that is closest to the current state of the original system and applies the control attached to this closest state over the associated holding time.

D. Complexity

The time complexity per iteration of the implementation in Algorithms 1-2 is $O\big(|S_n|^\theta (\log |S_n|)^2\big)$. The processing time from the beginning until the iMDP algorithm stops after n iterations is thus $O\big(|S_n|^{1+\theta} (\log |S_n|)^2\big)$. We note that when the procedure `ComputeTranProb` compute transition probabilities by solving a system of linear equations, the time complexity per iteration would be $O\big(|S_n|^\theta \log |S_n|\big)$, leading to total processing time $O\big(|S_n|^{1+\theta} \log |S_n|\big)$ [19]. The space complexity of the iMDP algorithm is $O(|S_n|)$ where $|S_n| = \Theta(n)$ due to our sampling strategy.

IV. Analysis

In this section, we present main results on the performance of the extended iMDP algorithm with brief proofs. More detailed proofs can be found in [27].

We first review the following key results of the approximating Markov chain method when no additional trajectory constraints are considered [28]. Local consistency implies the convergence of continuous-time interpolations of the trajectories of the controlled Markov chain to the trajectories of the stochastic dynamical system described by Eq. 1. Furthermore, the sequence of optimal cost-to-go of discrete MDPs converges uniformly to the optimal cost-to-go of the original problem. Previous results in [19] show that J_n returned from the iMDP algorithm converges pointwise to J^* in probability. That is, we are able to compute J^* in an incremental manner without directly computing J_n^* .

Now, in the presence of additional trajectory constraints, let $(\mathcal{M}_n = (S_n, U, P_n, G_n, H_n), \Delta t_n, J_n, C_n, \mu_n)$ denote the MDP, holding times, cost value function, constraint value function, and policy returned by Algorithm 1 at the end n iterations. As shown in [19], [27], the sequence of MDPs

$\{\mathcal{M}_n\}_{n=0}^{\infty}$ and holding times $\{\Delta t_n\}_{n=0}^{\infty}$ returned from the iMDP algorithm are locally consistent with the stochastic differential dynamics in Eq. 1 almost surely. We assume that optimal policies of the original continuous problem are obtainable[1]. The next theorem asserts the probabilistic soundness of the computed policies $\{\mu_n\}_{n=0}^{\infty}$ and the almost sure pointwise convergence of J_{n,μ_n} to J^*. We note that compared to previous results [19], while the uniform convergence of J_n to J^* happens in probability, the pointwise convergence of J_{n,μ_n} to J^* happens almost surely.

Theorem 1 *Let J_{n,μ_n} be the cost-to-go function of the returned policy μ_n on the discrete MDP $\mathcal{M}_n$. Similarly, let C_{n,μ_n} be the expected constraint value by executing the returned policy μ_n on the discrete MDP $\mathcal{M}_n$. Then, for all $z \in S_n$, we have*

$$\lim_{n\to\infty} |J_{n,\mu_n} - J^*(z)| = 0 \text{ w.p.1.}$$

Thus, for any $n \in \mathbb{N}$ and for any $z \in S_n$, $\{\mu_n(z)\}_{n=0}^{\infty}$ converges almost surely to $\mu^*(z)$ where μ^* is an optimal policy of the original continuous problem. Furthermore, for all $z \in S_n$:

$$\lim_{n\to\infty} |C_n(z) - C_{\mu^*}(z)| = 0 \text{ w.p.1,}$$
$$\lim_{n\to\infty} |C_{n,\mu_n}(z) - C_{\mu^*}(z)| = 0 \text{ w.p.1.}$$

As a corollary, $C_{\mu^*}(z) \in \Gamma$ w.p.1 for all $z \in \cup_{n=0}^{\infty} S_n$. That is, the sequence $\{\mu_n\}_{n=0}^{\infty}$ is probabilistically sound.

The almost sure pointwise convergence of J_{n,μ_n} to J^* can be proven similarly to Theorem 8 in [27]. The idea is that from any state $z \in S_n$, it is possible to construct a sequence of controls out of constructed controls from the procedure `ConstructControls` that converges in distribution to the optimal control process of the original continuous problem. The almost sure pointwise convergence of C_n and C_{n,μ_n} to C_{μ^*} can be seen as a special case of the above discussion where the control set at each $z \in S_n$ contains only one control $\mu_n(z)$.

The next theorem evaluates the quality of any-time control policies returned by Algorithm 3.

Theorem 2 *Let $\overline{\mu}_n : S \to U$ be the interpolated policy on S of $\mu_n : S_n \to U$ as described in Algorithm 3:*

$$z \in S: \ \overline{\mu}_n(z) = \mu_n(y_n) \text{ where } y_n = \text{argmin}_{z' \in S_n} ||z' - z||_2.$$

Then there exists an optimal control policy μ^* of the original problem so that for all $z \in S$:

$$\lim_{n\to\infty} \overline{\mu}_n(z) = \mu^*(z) \text{ w.p.1,}$$

if μ^* is continuous at z.

Proof: Fix $n \in \mathbb{N}$, for all $z \in S$, and $y_n = \text{argmin}_{z' \in S_n} ||z' - z||_2$, we have

$$\overline{\mu}_n(z) = \mu_n(y_n).$$

[1]Otherwise, an optimal relaxed control policy m^* exists [28].

By Theorems 1, we $\mu_n(y_n)$ converges to $\mu^*(y_n)$ almost surely where μ^* is an optimal policy of the original continuous problem. Thus, for all $\epsilon > 0$, there exists N such that for all $n > N$:

$$||\mu_n(y_n) - \mu^*(y_n)||_2 \leq \frac{\epsilon}{2} \text{ w.p.1.}$$

Under the assumption that μ^* is continuous at z, and due to $\lim_{n\to\infty} y_n = z$ almost surely, we can choose N large enough such that for all $n > N$:

$$||\mu^*(y_n) - \mu^*(z)||_2 \leq \frac{\epsilon}{2} \text{ w.p.1.}$$

From the above inequalities, for all $n > N$:

$$\begin{aligned} ||\mu_n(y_n) - \mu^*(z)||_2 \leq \ & ||\mu_n(y_n) - \mu^*(y_n)||_2 \\ & + ||\mu^*(y_n) - \mu^*(z)||_2 \leq \epsilon \text{ w.p.1.} \end{aligned}$$

Therefore,

$$\lim_{n\to\infty} ||\overline{\mu}_n(z) - \mu^*(z)||_2 = \lim_{n\to\infty} ||\mu_n(y_n) - \mu^*(z)||_2 = 0$$

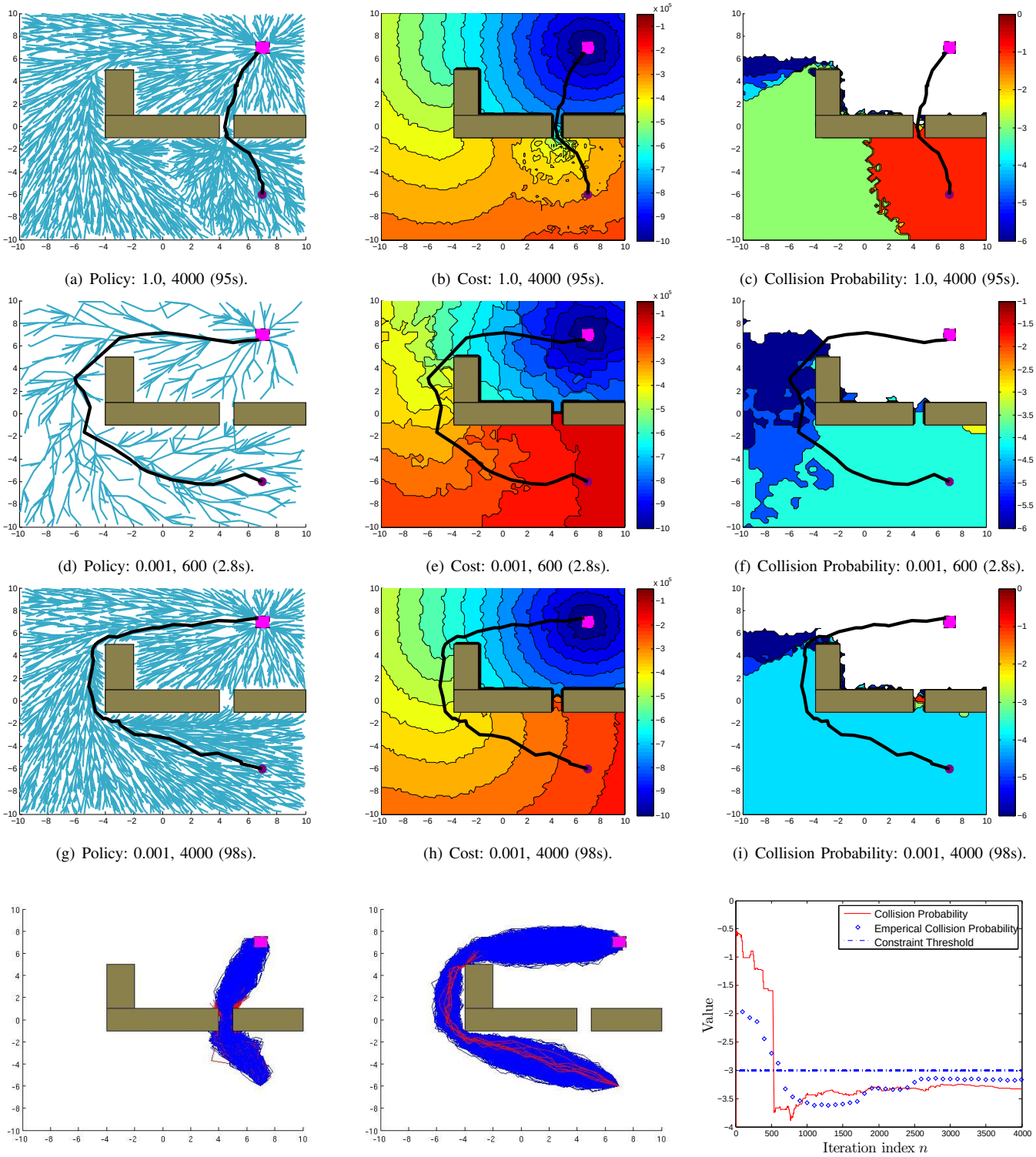
happens with probability one. ■

V. Experiments

In the following experiments, we used a computer with a 2.0-GHz Intel Core 2 Duo T6400 processor and 4 GB of RAM. We controlled a system with stochastic single integrator dynamics to a goal region with free ending time in a cluttered environment. The standard deviation of noise in each direction is 0.2. The system stops when it collides with obstacles. The cost function is the total energy spent to reach the goal, which is measured as the integral of square of control magnitude with discount rate $\alpha = 0.95$. The system pays the cost of -10^6 when reaching the goal region $\mathcal{X}_{goal}$. The maximum velocity of the system is one. The system stops when it collides with obstacles. At the same time, we considered the trajectory constraint that expresses the collision probability under the control policy (i.e. $\beta = 0.9999$, $r(x,u) = 0$ for all $x \in S, u \in U$, $k(x) = 1$ for $x \in \mathcal{X}_{obs}$ and $k(x) = 0$ otherwise). In this context, we often refer to *constraint values* as *probability value*.

We first set the upper value of the collision probability to 1.0, i.e. $\Gamma = (0, 1.0]$. Figures 1(a)-1(c) depict the policy, cost value function, constraint value function (in log scale) after $4,000$ iterations for this case. As we can see, the computed collision probability from the initial position is about 0.1, and the computed cost value for the initial position is about 4×10^{-5}. Since there is actually no constraint on the probability of collision with $\Gamma = (0, 1]$, the system takes risks going through the small gap between two obstacles to reach the goal as fast as possible.

In practice, we are interested in very small collision probability. Thus, we then set $\Gamma = (0, 0.001]$, which allows for the maximum tolerated collision probability 0.1%. As above, Figs. 1(d)-1(f) show the policy, cost value function, constraint value function after 600 iterations iterations respectively after about 2.8 seconds. From the plots, under the policy returned by the algorithm, at the initial position, the computed cost value is about 1×10^{-6}, and the computed collision probability is 0.0003. To achieve this low risk, the system takes



(a) Policy: 1.0, 4000 (95s).

(b) Cost: 1.0, 4000 (95s).

(c) Collision Probability: 1.0, 4000 (95s).

(d) Policy: 0.001, 600 (2.8s).

(e) Cost: 0.001, 600 (2.8s).

(f) Collision Probability: 0.001, 600 (2.8s).

(g) Policy: 0.001, 4000 (98s).

(h) Cost: 0.001, 4000 (98s).

(i) Collision Probability: 0.001, 4000 (98s).

(j) Empirical trajectories for Fig. 1(a) (8.2%).

(k) Empirical trajectories for Fig. 1(g) (0.07%).

(l) Computed and tested collision probabilities.

Fig. 1. A system with stochastic single integrator dynamics in a cluttered environment. The cost function is the total energy spent to reach the goal, which is measured as the integral of square of control magnitude. The trajectory constraint expresses the probability of collision (with discount rate $\beta = 0.9999$). Figures 1(a)-1(c) depict the policy, cost value function, constraint value function (in log scale) after $4,000$ iterations when the upper bound of collision probability is $1.0(100\%)$. The first number in the title is the constraint upper bound, and the second number is the number of iterations. Similarly, Figures 1(d)-1(f) and Figures 1(g)-1(i) show the corresponding plots for the constraint upper bound $0.001(0.1\%)$ after 600 iterations and $4,000$ iterations respectively. Figure 1(j) shows 10,000 empirical trajectories for the returned policy in Fig. 1(a). Collision-free trajectories are plotted in green, and colliding trajectories are plotted in red. The empirical collision probability is 8.2%. Figure 1(k) shows 10,000 empirical trajectories for the returned policy in Fig. 1(g) with the resulting empirical collision probability 0.07%. When $\Gamma = (0, 0.001]$, in Fig. 1(l), constraint value function, constraint threshold, and empirical collision probability over iterations are plotted on a semi-log graph where values are averaged from 50 trials. In each trial, empirical collision probability is obtained using 10,000 tested trajectories and is plotted for every 100 iterations.

a longer route that stays away from the obstacles. Similarly, Fig. 1(g)-1(i) present the corresponding plots after 4000 iterations. As we can see, the computed collision probability (0.000938) for the initial position increases to allow for smaller cost value (-2.8×10^{-5}) from the starting location.

Finally, we tested the empirical collision probability of the returned policies compared to the computed probability value. Figure 1(j) shows $10,000$ empirical trajectories for the returned policy in Fig. 1(a) when $\Gamma = (0, 1.0]$ where the empirical collision probability is 0.082. Similarly, Fig. 1(k) shows $10,000$ empirical trajectories for the returned policy in Fig. 1(g) when $\Gamma = (0, 0.001]$ with the resulting empirical collision probability 0.0007. Furthermore, when $\Gamma = (0, 0.001]$, we compare empirical collision probabilities and computed collision probability from the initial position over iterations on a semi-log graph in Fig. 1(l). In this plot, values are averaged from 50 trials, and in each trial, empirical collision probability is obtained using $10,000$ tested trajectories. As we can see, the computed collision probability approximates very well the actual collision probability when we execute the returned policies. This observation agrees with the probabilistic soundness property of the algorithm.

VI. Conclusions

We have introduced and analyzed the extension of the incremental Markov Decision Process (iMDP) algorithm for stochastic optimal control in the presence of additional trajectory constraints. In particular, trajectory constraints represent different aspects of controlled trajectories in terms of expected costs. Thus, we solve the stochastic optimal control with bounded collision probability as a special case. The algorithm inherits the efficient computation from iMDP to compute the expected costs using asynchronous policy evaluations and value iterations. In addition, the algorithm guarantees the probabilistic soundness and asymptotic optimality of computed feedback policies as the number of iterations approaches infinity. In our future work, we plan to implement the algorithm outlined in this paper on robotic platforms for practical demonstration.

Acknowledgments

This research was supported in part by the National Science Foundation, grant CNS-1016213, and by the Army Research Office, MURI grant W911NF-11-1-0046.

References

[1] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. on Control Systems Technologies*, vol. 17, no. 5, pp. 1105–1118, 2009.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*, 2001.

[3] W. H. Fleming and J. L. Stein, "Stochastic optimal control, international finance and debt," *Journal of Banking and Finance*, vol. 28, pp. 979–996, 2004.

[4] S. P. Sethi and G. L. Thompson, *Optimal Control Theory: Applications to Management Science and Economics*, 2nd ed. Springer, 2006.

[5] E. Todorov, "Stochastic optimal control and estimation methods adapted to the noise characteristics of the sensorimotor system," *Neural Computation*, vol. 17, pp. 1084–1108, 2005.

[6] R. Alterovitz, T. Simon, and K. Goldberg, "The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty," in *in Robotics: Science and Systems III (Proc. RSS 2007*. MIT Press, 2008, pp. 246–253.

[7] V. D. Blondel and J. N. Tsitsiklis, "A survey of computational complexity results in systems and control," *Automatica*, vol. 36, no. 9, pp. 1249–1274, 2000.

[8] C. Chow and J. Tsitsiklis, "An optimal one-way multigrid algorithm for discrete-time stochastic control," *IEEE Transactions on Automatic Control*, vol. AC-36, pp. 898–914, 1991.

[9] R. Munos, A. Moore, and S. Singh, "Variable resolution discretization in optimal control," in *Machine Learning*, 2001, pp. 291–323.

[10] L. Grne, "An adaptive grid scheme for the discrete hamilton-jacobi-bellman equation," *Numerische Mathematik*, vol. 75, pp. 319–337, 1997.

[11] S. Wang, L. S. Jennings, and K. L. Teo, "Numerical solution of hamilton-jacobi-bellman equations by an upwind finite volume method," *J. of Global Optimization*, vol. 27, pp. 177–192, November 2003.

[12] M. Boulbrachene and B. Chentouf, "The finite element approximation of hamilton-jacobi-bellman equations: the noncoercive case," *Applied Mathematics and Computation*, vol. 158, no. 2, pp. 585–592, 2004.

[13] J. Rust, "Using Randomization to Break the Curse of Dimensionality,," *Econometrica*, vol. 56, no. 3, May 1997.

[14] ——, "A comparison of policy iteration methods for solving continuous-state, infinite-horizon markovian decision problems using random, quasi-random, and deterministic discretizations," 1997.

[15] L. E. Kavraki, P. Svestka, L. E. K. P. Vestka, J. claude Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 566–580, 1996.

[16] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Tech. Rep., 1998.

[17] Karaman and Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, June 2011.

[18] J. Kim and J. P. Ostrowski, "Motion planning of aerial robot using rapidly-exploring random trees with dynamic constraints," in *ICRA*, 2003, pp. 2200–2205.

[19] V. A. Huynh, S. Karaman, and E. Frazzoli, "An incremental sampling-based algorithm for stochastic optimal control," in *ICRA*, 2012, pp. 2865–2872.

[20] J. N. Tsitsiklis, "Efficient algorithms for globally optimal trajectories," *IEEE Transactions on Automatic Control*, vol. 40, pp. 1528–1538, 1995.

[21] P. Kosmol and M. Pavon, "Lagrange approach to the optimal control of diffusions," *Acta Applicandae Mathematicae*, vol. 32, pp. 101–122, 1993, 10.1007/BF00998149.

[22] ——, "Solving optimal control problems by means of general lagrange functionals," *Automatica*, vol. 37, no. 6, pp. 907 – 913, 2001.

[23] D. E. Kirk, *Optimal Control Theory: An Introduction*. Dover Publications, Apr. 2004.

[24] L. Blackmore, M. Ono, A. Bektassov, and B. C. Williams, "A probabilistic particle-control approximation of chance-constrained stochastic predictive control," *IEEE Transactions on Robotics*, vol. 26, no. 3, 2010.

[25] A. G. Banerjee, M. Ono, N. Roy, and B. C. Williams, "Regression-based LP solver for chance-constrained finite horizon optimal control with nonconvex constraints," in *Proceedings of the American Control Conference*, San Francisco, CA, 2011.

[26] J. Steinhardt and R. Tedrake, "Finite-time regional verification of stochastic nonlinear systems," in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.

[27] V. A. Huynh, S. Karaman, and E. Frazzoli, "An incremental sampling-based algorithm for stochastic optimal control," *arXiv:1202.5544v1 [cs.RO]*, 2012.

[28] H. J. Kushner and P. G. Dupuis, *Numerical Methods for Stochastic Control Problems in Continuous Time (Stochastic Modelling and Applied Probability)*. Springer, Dec. 2000.

[29] I. Karatzas and S. E. Shreve, *Brownian Motion and Stochastic Calculus (Graduate Texts in Mathematics)*, 2nd ed. Springer, Aug. 1991.