Sampling-based Algorithms for Stochastic Optimal Control

by

Vu Anh Huynh

M.S., Massachusetts Institute of Technology (2008)
B.Eng., Nanyang Technological University (2007)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2014

Signature redacted

Author ....
Department of Aeronautics and Astronautics
May 16, 2014

Signature redacted

Certified by ....
Emilio Frazzoli
Professor of Aeronautics and Astronautics
Thesis Supervisor

Signature redacted

Certified by ....
Nicholas Roy
Associate Professor of Aeronautics and Astronautics

Signature redacted

Certified by ....
John N. Tsitsiklis
Professor of Electrical Engineering and Computer Science

Signature redacted

Accepted by ....
Paulo C. Lozano
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

Sampling-based Algorithms for Stochastic Optimal Control

by
Vu Anh Huynh

Submitted to the Department of Aeronautics and Astronautics
on May 16, 2014, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Controlling dynamical systems in uncertain environments is fundamental and essential in several fields, ranging from robotics, healthcare to economics and finance. In these applications, the required tasks can be modeled as continuous-time, continuous-space stochastic optimal control problems. Moreover, risk management is an important requirement of such problems to guarantee safety during the execution of control policies. However, even in the simplest version, finding closed-form or exact algorithmic solutions for stochastic optimal control problems is computationally challenging.

The main contribution of this thesis is the development of theoretical foundations, and provably-correct and efficient sampling-based algorithms to solve stochastic optimal control problems in the presence of complex risk constraints.

In the first part of the thesis, we consider the mentioned problems without risk constraints. We propose a novel algorithm called the incremental Markov Decision Process (iMDP) to compute incrementally *any-time* control policies that approximate arbitrarily well an optimal policy in terms of the expected cost.. The main idea is to generate a sequence of finite discretizations of the original problem through random sampling of the state space. At each iteration, the discretized problem is a Markov Decision Process that serves as an incrementally refined model of the original problem. We show that the iMDP algorithm guarantees asymptotic optimality while maintaining low computational and space complexity.

In the second part of the thesis, we consider risk constraints that are expressed as either bounded trajectory performance or bounded probabilities of failure. For the former, we present the first extended iMDP algorithm to approximate arbitrarily well an optimal feedback policy of the constrained problem. For the latter, we present a martingale approach that diffuses a risk constraint into a martingale to construct time-consistent control policies. The martingale stands for the level of risk tolerance that is contingent on available information over time. By augmenting the system dynamics with the martingale, the original risk-constrained problem is transformed into a stochastic target problem. We present the second extended iMDP algorithm to approximate arbitrarily well an optimal feedback policy of the original problem by sampling in the augmented state space and computing proper boundary values for the reformulated problem. In both cases, sequences of policies returned from the extended algorithms are both probabilistically sound and asymptotically optimal.

The effectiveness of these algorithms is demonstrated on robot motion planning and control problems in cluttered environments in the presence of process noise.

Thesis Supervisor: Emilio Frazzoli
Title: Professor of Aeronautics and Astronautics

Acknowledgments

This thesis would not have been possible without the guidance, support, and help of many individuals to whom I want to express my sincere gratitude here.

First and foremost, my greatest appreciation goes to my thesis supervisor, Prof. Emilio Frazzoli, for his vision and guidance. He is truly a knowledgeable and insightful supervisor, who has always inspired me with his creativity and challenged me with his intelligent questions. Most importantly, he has provided me with the freedom to explore and work on fundamental research problems. He also gave me precious advice on how to manage work-life balance from his personal experience, which made my life at MIT so much easier.

I am also grateful to the other members of my thesis committee, Prof. Nicholas Roy and Prof. John Tsitsiklis, for their valuable feedback to complete this thesis. Especially, Prof. Nicholas Roy introduced me to the world of stochastic modeling and control in robotics seven years ago when I worked with him during my Master studies. Thanks to his introduction, I had decided to continue exploring open problems in the field at the doctorate level. I have also learned tremendously from the writings of Prof. John Tsitsiklis. His writings have deepened my knowledge in many ways.

I would like to thank Marco Pavone and Sertac Karaman for their friendships and their time as readers of this thesis. I knew of Prof. Emilio Frazzoli's supervision style through Marco, and I am glad that I have made the right choice. My collaboration with Sertac during the first phase of this thesis was invaluable. Much of the development of the basic algorithms was shaped during our fruitful discussions in the first year.

My sincere thanks go to Prof. Leonid Kogan for his help during my adventure into Financial Economics and his valuable suggestions for the second part of this thesis. Prof. Leonid Kogan has connected me to a vibrant research community in his field that studies similar research problems. It is very satisfying for me to discover and combine knowledge and tools in several fields in order to tackle my research questions. I am also thankful to Prof. Michael Eichmair for his dedicated teaching. His thought-provoking lectures helped me to grasp the fundamentals of measure theory, which are very essential for carrying out this research.

I am grateful to a fantastic group of professors, colleagues, and friends at the department of Aeronautics and Astronautics, the Laboratory for Information and Decision Systems (LIDS), and the Aerospace Robotics and Embedded Systems (ARES) group for sharing their interesting research topics and initiative ideas through numerous seminars, talks, and weekly group meetings. I would also like to thank the administrative staff in my department and LIDS for their assistance in the last five years. Especially, many thanks go to my labmates at ARES who have made my journey more enjoyable.

This thesis is lovingly dedicated to my father, Huynh Van Xuan, and my mother, Ho Thi Tuyet Ngoc for their support, encouragement, and constant love throughout my life. My father has always been my source of inspiration and has taught me the value of knowledge and education. Many of my successes today are the result of his vision and teaching. My mother has sacrificed most of her life for my education.

Since my childhood, due to my father's illness, my mother has worked so hard as a household breadwinner so that my father and I could concentrate on my studies. Until today, she is again sacrificing her time to fly to the U.S. and help me to take care of my newborn baby during the first two years of her life. My deep appreciation also goes to my parents-in-law, Truong Thi Minh Hanh and Bui Thien Lam, for arranging their precious time to join my mother in taking care of my daughter in the U.S. To my sisters and brothers, Nhung, Ngan, Phuoc, I would like to thank you for your love and understanding while our mothers are far away from home. Without the endless support from my family, I would have not been able to finish this thesis.

My most special thanks go to my partner and also my colleague, Kim Thien Bui, for her never-ending love, encouragement, and sympathy. She has been with me since primary school to expand our knowledge together. Words really cannot articulate how important your support is for me to finish this work. You have been very patient to listen to my research ideas and to share the joy and the sadness of the up and down moments in my research. Thank you for always being by my side and taking care of every aspect of my life from the very early stage of this research. This work is also dedicated to our daughter, Mai (Misa) Huynh. I hope that my work would provide you with a platform to achieve more than what I can now, and I look forward to watching you grow.

Finally, I would like to thank Dr. Arthur Gelb and Ms. Linda Gelb for the generous financial support during the first two years of my Ph.D. studies. The Gelb family's support has provided me with the academic freedom to pursue my intellectual curiosity. This work was also supported by the National Science Foundation through grant CNS-1016213, the Army Research Office MURI through grant W911NF-11-1-0046. Their support is gratefully acknowledged. I am also thankful to the Singapore-MIT Alliance for Research and Technology Center (SMART) for hosting me during the summers from 2010 to 2012 when I conducted part of my research there.

Contents

1 **Introduction** **13**
- 1.1 Stochastic Optimal Control 13
- 1.2 Robot Motion Planning 16
- 1.3 Risk Management 20
- 1.4 Statement of Contributions 23
- 1.5 Outline 25

2 **Background and Preliminaries** **27**
- 2.1 Basic Definitions and Notations 27
- 2.2 Probabilistic Models 31
  - 2.2.1 Brownian motion 31
  - 2.2.2 Controlled diffusion processes 34
  - 2.2.3 Geometric dynamic programming 36
  - 2.2.4 Markov chains 37
- 2.3 K-Nearest Neighbor Graphs 38

3 **Stochastic Optimal Control: Formulation and Algorithm** **41**
- 3.1 Problem Formulation 41
- 3.2 The Markov Chain Approximation Method 44
- 3.3 Incremental Markov Decision Process (iMDP) Algorithm 50
  - 3.3.1 Primitive procedures 50
  - 3.3.2 iMDP algorithm description 53
  - 3.3.3 Complexity of iMDP 57
  - 3.3.4 Feedback control 58
- 3.4 Analysis 59
- 3.5 Experiments 61
- 3.6 Proofs 67
  - 3.6.1 Proof of Theorem 3.4.1 68
  - 3.6.2 Proof of Lemma 3.4.2 69
  - 3.6.3 Proof of Lemma 3.4.3 70
  - 3.6.4 Proof of Theorem 3.4.5 70
  - 3.6.5 Proof of Theorem 3.4.6 80
  - 3.6.6 Proof of Theorem 3.4.7 84

4 **Stochastic Control with trajectory Performance Constraints** **85**
4.1 Problem Formulation 85
4.2 Extended iMDP Algorithm 86
4.3 Analysis 89
4.4 Experiments 90

5 **Martingale Approach for Risk Management** **95**
5.1 Problem Formulation 96
5.2 Martingale approach 97
5.2.1 Diffusing risk constraints 97
5.2.2 Stochastic target problem 98
5.2.3 Characterization and boundary conditions 100
5.3 Algorithm 103
5.3.1 Markov chain approximation 103
5.3.2 Extended iMDP algorithm 106
5.4 Analysis 112
5.5 Experiments 114

6 **Conclusions** **125**
6.1 Summary 125
6.2 Future Directions 126
6.2.1 Partially-observable states 126
6.2.2 Error bounds for anytime policies 127
6.2.3 Investigation of new noise models 127
6.2.4 Logic constraints 128
6.2.5 Exploiting dynamics of mechanical systems 128
6.2.6 Parallel implementation 128
6.2.7 Collaborative multi-agent systems 129
6.2.8 Stochastic games 129
6.2.9 Exploring applications in management and finance 129

List of Figures

3-1 An illustration of a continuous-time interpolation. 47
3-2 An illustration of transition probability construction. 52
3-3 Steps of the iMDP algorithm. 56
3-4 An illustration of Markov chains over iterations. 57
3-5 Results of iMDP on a stochastic LQR problem. 62
3-6 An operating environment for the second experiment. 63
3-7 A system with stochastic single integrator dynamics in a cluttered environment. 64
3-8 Markov chains over iterations. 65
3-9 Performance against different process noise magnitude. 66
3-10 Results of a 6D manipulator example. 67
3-11 An illustration for Lemma 3.6.1. 68
3-12 A realization of the random sequence B_n. 74
3-13 An illustration for Lemma 3.6.5. 82

4-1 An example of bounded trajectory performance. 91
4-2 Bounded trajectory performance results. 92

5-1 A domain of $\mathcal{OPT}4$ 101
5-2 An illustration of failure probabilities due to optimal control policies of the unconstrained problem. 102
5-3 A feedback-controlled trajectory of $\mathcal{OPT}3$ 112
5-4 A modified feedback-controlled trajectory of $\mathcal{OPT}3$ 113
5-5 An operating environment for the single integrator system. 115
5-6 Anytime policies and Markov chains in S. 116
5-7 Boundary values. 117
5-8 Anytime policies and Markov chains in $\overline{S}$. 118
5-9 Examples of incremental value functions over iterations. 119
5-10 Examples of trajectories from policies of the unconstrained problem and min-collision probability problem. 120
5-11 An example of controlled trajectories using boundary values. 121
5-12 Trajectories after 5000 iterations starting from $(6.5, -3)$. 122
5-13 Failure ratios for the first N trajectories ($1 \leq N \leq 5000$) starting from $(6.5, -3)$ with different values of η. 123

List of Tables

3.1 Comparison of sampling-based algorithms. 58

5.1 Failure ratios and average costs for 5000 trajectories for Fig. 5-13. . . 119

Chapter 1

Introduction

Stochastic optimal control is a subfield of control theory that considers mathematical models with uncertainty in the control process. The goal of stochastic optimal control is to design feedback policies that perform desired tasks with minimum costs despite the presence of noises. Historically, mathematical models used in stochastic optimal control are often derived for engineering systems such as mechanical systems. Due to their abstraction, these models are also applied for problems in other domains such as mathematical economics [1,2] and mathematical finance [3]. Therefore, stochastic optimal control has been studied extensively by several research communities, and each community focuses on different theoretical and implementation aspects of the field. Researchers also find applications of stochastic optimal control in diverse fields ranging from robotics [4], biology [5], healthcare [6] to management science, economics and finance [7,8].

In this thesis, we primarily focus on applications of stochastic optimal control in robotics, especially the problem of robot motion planning and control. In recent years, several advanced autonomous systems have been built to operate in uncertain environments such as Mars rovers for planetary missions [9], autonomous cars providing urban mobility on demand [10, 11], and small aerial vehicles operating in the presence of stochastic wind [12]. In many of these applications, the systems operate in worlds that are inherently continuous in time and space under a continuous control space. Moreover, we are often concerned with several aspects of the control process. For example, in Mars exploration missions, we want that a Mars rover departs from an origin to reach a destination with minimum energy and at the same time minimizes the risk of failure. Therefore, in this thesis, we consider a broad class of continuous-time, continuous-space stochastic optimal control problems that may contain additional complex constraints. We aim to provide a generic sampling-based approach to construct incremental solutions for the mentioned problems.

1.1 Stochastic Optimal Control

Informally speaking, given a system with dynamics specified by a *controlled diffusion process* with a state space and a control space that describe an operating environment and available controls, a stochastic optimal control problem is to find an optimal feedback policy to minimize the expected cost-to-go known as an objective function. The resulting objective function evaluated at an optimal feedback policy is called an *optimal cost-to-go function* or a *value function*. In certain applications of interest, we can also have risk constraints that are expressed as bounded trajectory performance or bounded probabilities of failure. A large body of literature is devoted to characterize and provide *numerical* and *algorithmic* solutions to these problems and their related versions from multiple perspectives, which will be briefly described in this section.

Optimal solution characterization

The control community has concentrated on characterizing optimal solutions to stochastic control problems. Since 1950, a variety of different approaches to stochastic optimal control have been investigated. Arguably, *dynamic programming* proposed by Bellman in 1957 [13] is one of the most well-known approaches. The dynamic programming principle provides a proper way to relate time-t optimal value function to any later time-τ optimal value function. Bellman's principle of optimality leads to nonlinear partial differential equations (PDEs) of second order, known as Hamilton-Jacobi-Bellman (HJB) equations, whose solutions, if exist, are shown to be the value functions of control problems.

Following Bellman, several works focus on finding conditions under which HJB equations have solutions (see survey in [14–18]). Establishing such conditions often limits the class of problems that can be handled by the dynamic programming approach [19]. In particular, these conditions allow value functions to be smooth enough so that they satisfy HJB equations in the classical or usual sense. However, in practice, value functions are often not smooth enough to be classical solutions. On the other hand, there are many functions other than value functions satisfying HJB equations almost everywhere.

Thus, intensive research efforts have focused on new solution concepts that allow for non-smooth value functions. Since 1983, *viscosity solutions* have gained popularity as an alternative and natural solution concept for HJB equations [17, 20]. Viscosity solutions are a weak formulation of solutions to HJB equations that enables us to continue the dynamic programming approach. For a large class of optimal control problems, the value function is the unique viscosity solution of the associated HJB equation. However, for several problems with complex constraints, deriving the associated HJB equations from the dynamic programming principle encounters technical difficulties related to the measurable selection argument. Recently, in 2011, an approach called *weak dynamic programming* was proposed by Bouchard and Touzi [21] to derive HJB equations and find viscosity solutions that can avoid measurability issues. As shown in the authors' very recent works, the weak dynamic programming approach enables us to establish the HJB equation for a broader class of interesting problems with terminal state constraints [22–26].

Indeed, deriving HJB equations for different classes of problems is still an on-going and active research topic. While deriving HJB equations is the utmost research goal for characterizing classical or viscosity optimal solutions, computing a solution of a stochastic optimal control problem in an efficient way is a crucial research question in practice. In the following, we discuss the computational complexity and methods to solve stochastic control problems.

Computational complexity

Unfortunately, general continuous-time, continuous-space problems do not admit closed-form or exact algorithmic solutions and are known to be computationally challenging (see, e.g., [27–31]). Problems with closed-form solutions such as the linear quadratic gaussian (LQG) problem [17,32,33] or Merton's portfolio problem [34] are rare. These exceptions are due to special problem structures such as a quadratic value function for the LQG problem or an optimal constant-fraction investment strategy for Merton's portfolio problem.

General continuous time and space problems can be solved approximately by discretizing time and space [27]. This discretization is used in numerical methods that solve HJB equations or in the construction of approximating discrete-time finite-state Markov Decision Processses (MDPs). Discrete-time finite-state MDP problems can be solved, e.g., by linear programming, in time which increases polynomially in the number of states. However, to obtain a good approximation, we often need a large number of states. This leads to the phenomenon called "curse of dimensionality" in which both required storage space and running time increase exponentially in the dimension of the space [27]. In practice, discretization is only considered computationally feasible up to five-dimensional state spaces.

The above result strongly suggests that the complexity of finding asymptotically-optimal solutions of continuous-time continuous-space stochastic optimal control problems grows exponentially in the dimension of the state space.

Numerical and algorithmic methods

In the light of the above complexity result, several works have focused on computing *approximate solutions* to stochastic optimal control problems. A popular approach is to compute solutions to HJB equations numerically (see, e.g., [35–37]). However, for new classes of problems with complex constraints, deriving the HJB equations is often hard. In addition, for problems such as singular stochastic control and impulsive control, the HJB equations are in fact a system of partial differential inequalities. The existence, uniqueness of viscosity solutions and regularity theory for this class of PDEs are not well understood [38].

Thus, other methods approximate a continuous problem with a *single* discrete-time finite-state Markov Decision Process (MDP) [39,40] without invoking the associated HJB equation. When dealing with finite-state MDPs, we can use *specialized algorithms* such as policy iteration, value iteration and their parallel versions to find ϵ-optimal solutions. The thorough treatment of these algorithms can be found in the work by Bertsekas and Tsitsiklis [41,42]. However, having a single MDP to approximate the continuous problem often looses the fidelity of the original continuous problem model. Furthermore, assigning ad-hoc transition probabilities on the MDP can lead to inconsistent approximation. The method described below, pioneered by Kushner and Dupuis, addresses several drawbacks of the previous methods.

For the last three decades, Kushner, Dupuis, and their colleagues have developed a powerful method called *Markov chain approximation* (see, e.g., [43, 44] and references therein) to compute numerically value functions of a wide spectrum of stochastic optimal control problems. Conceptually, the Markov chain approximation method constructs *a sequence* of MDPs to consistently approximate the underlying continuous dynamics. This probabilistic approach, which relies on the theory of weak convergence [45 47], offers several advantages. First, the method does not require smooth value functions and does not derive the associated HJB equations. This advantage is significant for problems where the PDE theory for the associated HJB equations is difficult to tackle. Second, the method uses physical insights of the systems to construct the approximation. Interesting, given an elliptic PDE, it is possible to construct reversely an artificial stochastic dynamics for the equation [24,44]. Hence, Markov chain approximation is a probabilistic method to compute solutions of elliptic equations as well. Third, the method provides a mild sufficient condition, called *local consistency*, to construct consistent approximations. As local consistency can be constructed easily in most cases, it is also straightforward to implement the method.

Thus far, the above methods can be classified as *deterministic methods.* As discussed above, due to discretization, the complexity these deterministic algorithms, however, scales exponentially with the dimension of the state and control spaces. Moreover, the above algorithms require global strategies to devise such *a priori* discretization, which becomes difficult to manage and automate for high dimensional state spaces. For robotics applications where the state spaces are often unstructured due to cluttered environments or even unknown and dynamic, such global strategies are undesirable.

Remarkably, as noted in [27,48,49], algorithms based on random (or quasi-random) sampling of the state space, also known as *sampling-based algorithms*, provide a possibility to alleviate the curse of dimensionality when the control inputs take values from a finite set. Nevertheless, designing sampling-based algorithms for stochastic optimal control remains largely unexplored. At the same time, sampling-based algorithms can also be traced back to research in (deterministic) motion planning [50–52] in robotics and related disciplines such as computational biology, computer animation [53 58]. This field of research has been conducted in parallel with the stochastic optimal control research in the last three decades. In the following section, we will review the development of the field, which will shed light on a better method for discretization.

1.2 Robot Motion Planning

As robots become an integral part of industry and daily life, the *(deterministic) robot motion planning problem* has received much attention from the robotics and automation research community. Given a robot with continuous-time dynamics operating in a *noise-free* environment, an initial state in a continuous configuration space.[1], a set of goal states, the robot motion planning problem is to find a sequence of *feasible* control inputs to drive the system from the initial state to one of the goal states and at the same time avoid collision with obstacles (see, e.g., [52, 59, 60] and references therein). The optimal version of the problem called *optimal motion planning* seeks for a feasible solution that minimizes some performance measure. These problems, which can be cast as (deterministic) (optimal) control problems, have mathematical formulations that are closely related to the stochastic optimal control formulations considered in this thesis.

The motion planning problem is known to be computationally hard, and the basic version called the generalized piano movers is proven to be PSPACE–hard[2] in the dimension of the configuration space by Reif in 1979 [61]. In addition, in 1988, Canny showed that computing the shortest path in a three-dimensional Euclidean space populated with obstacles is NP–hard in the number of obstacles [62]. Therefore, the optimal motion planning is computationally challenging even when the dimension of the configuration space is fixed. As the optimal motion planning problem can be thought as a"stochastic" optimal control problem with negligible noise magnitude, these results further assert the computational challenges involved in solving stochastic optimal control problems.

The PSPACE and NP complexity classes make *complete and exact algorithms* for motion planning, which return a valid solution in finite time, if one exists, and failure otherwise, unsuitable for practical usage [63–65]. The first practical approach called cell decomposition methods [66] provides *resolution completeness*, which means a valid solution, if one exists, is returned when the resolution parameters are set fine enough. The second practical approach called potential fields [67] provides complete solutions by using appropriate navigation functions. Although the two approaches can be applied to problems with state spaces of up-to five dimensions, cell decomposition methods suffer from the curse of dimensionality due to the large number of cells and difficult cell management [68], and potential field methods suffer from local minima [69]. More importantly, all previously mentioned methods require an explicit representation of the obstacles in the configuration space for the construction of solutions. Hence, these methods are not suitable for high dimensional state spaces and environments with a large number of obstacles.

Sampling-based algorithms

Therefore, to overcome the above difficulties, a class of *sampling-based algorithms* for the motion planing problem have been studied since the 1990s [50, 70–75]. The main advantage of these algorithms is to avoid such an explicit representation of obstacles in the configuration space by using feasibility tests of candidate trajectories. This leads to significant computational savings for problems with high dimensional

[1] The configuration space of a robot is identical to the state space if the robot is purely kinematic.

[2] PSPACE complexity class includes decision problems for which answers can be found with memory which are polynomial in the size of the input. The run time is not constrained. It is believed that NP class is proper subset of PSPACE class.

state spaces in cluttered environments. Instead of providing completeness guarantees, these algorithms provide *probabilistic completeness* in the sense that the probability of failing to return a solution, if one exits, decays to zero as the number of samples approaches infinity [76–83].

One of the first and most popular sampling-based algorithms is the Probabilistic RoadMap (PRM) algorithm proposed by Kavraki et al. [50,77]. The PRM algorithm first constructs an *a priori* graph, known as the roadmap, representing a rich set of collision-free trajectories and then answers multiple online queries by computing the shortest paths that connect initial states and final states through the roadmap.

While the PRM algorithm is suitable for environments such as factory floors where the roadmap is needed to build once, most applications only require a single query as the robot moves from one environment to another unknown environment. Moreover, computing the roadmap *a priori* may be computationally demanding. Thus, an incremental sampling-based algorithm called the Rapidly-Exploring Random Tree (RRT) were proposed by LaValle and Kuffner to avoid the need to specify *a priori* samples and tailored for single-query motion planning applications [51,84,85].

The RRT algorithm constructs a tree-based structure connecting an initial state to a goal region, which efficiently searches non-convex high dimensional search spaces. The algorithm is designed to determine (i) which node of the tree needs to be expanded, and (ii) in which direction the tree should explores. To achieve this, the algorithm picks a *random* configuration state and chooses a node in the tree to expand that is closest to the random state in terms of a Euclidean distance. Then, from the closest expanding node, the algorithm simulates the robot dynamics under some control inputs towards the random state so that the extended node is as close as possible to the random state. If the resulting trajectory is collision-free, it is feasible and added to the tree. As a result, the RRT algorithm chooses an expanding node that is proportional to the size of its Voronoi region and tends to grow towards large unsearched areas.

Several variants of the RRT algorithm have been studied extensively [78,85 92] and shown to work very well for systems with nonlinear differential dynamics [71,78]. The algorithm has also been implemented on several robotic platforms [10,93–96]. We emphasize that besides avoiding an explicit representation of obstacles in the configuration space, the RRT algorithm has a very simple scheme to manage its data structure in a large search space.

Sampling-based RRT-like algorithms can be implemented efficiently using the following primitive procedures of reduced complexity: random sampling, k-nearest neighbors search, local steering, collion-checking, and local node processing. Although the specific implementation of these primitive procedures in different RRT-like algorithms may differ slightly, the overall structure of these algorithms remain the same. Recent work by Bialkowski et at. [97–100] exploits the interconnection of these primitive procedures to optimize and significantly reduce the running time of RRT-like algorithms.

Despite practical successes of the RRT algorithm, the quality of the returned path and insights into the structure of constructed trees received little attention before a recent work by Karaman and Frazzoli in 2011 [52]. In this work, the authors have shown that the RRT algorithm fails to converge to optimal solutions with probability one and have also proposed the RRT* algorithm which guarantees almost-sure convergence to globally optimal solutions. The RRT* algorithm "rewires" the tree as it discovers new lower-cost paths reaching the nodes that are already in the tree. It is shown that the asymptotic computational complexity of the RRT* algorithm is essentially the same as that of RRTs. The authors analyze the problem using tools in the theory of random geometric graphs, which provides better understanding of the structure of random trees.

The theory of *random geometric graphs* is rich (see, e.g., [101–106]). Random geometric graphs are defined as stochastic collections of points in the metric space connected by edges when certain conditions are met. Depending on the conditions to connect edges, we have different random graph models. For instance, when an edge is formed if the distance between the two points is bounded by a positive constant, we have Gilbert's disc model [101]. Another popular model called k-nearest neighbor graph considers edges between k nearest neighbors [103]. A remarkable result in this field certifies that when $k = O(\log n)$ where n is the number of points, the resulting graph is connected asymptotically almost surely and thus has optimal shortest paths in the limit as the number of points approaches infinity. This result is sharp in the sense that fewer connections than this rate are almost surely sub-optimal.

It turns out that the above result plays a significant role in analyzing the RRT* algorithm. From the analysis in [52], it is clear that the success of RRT* algorithm for online robotic optimal motion planning applications in cluttered environments are due to two main features of the algorithm. First, the construction of random trees and the processing of optimal cost can be handled *locally* for each newly added sample. Second, despite that local processing, desirable *global* properties such as connectivity and optimality are still guaranteed in a suitable probabilistic sense. From the above discussion, we observe that constructing such random graphs and random trees in RRT-like algorithms is a randomized method to perform incremental discretization or cell decomposition of the configuration space. This observation suggests that randomized methods would offer similar benefits in handling stochastic optimal control problems.

Nevertheless, RRT-like algorithms are not suitable for the purpose of stochastic optimal control. In particular, RRT-like algorithms compute open-loop plans in the obstacle-free space, and during the execution phase, the robot must perform exact point-to-point steering to traverse from an initial state to a goal region. Hence, these algorithms are not aware of inherent uncertainty in system dynamics even when the robot constantly re-plans after being out of its open-loop plans due to the underlying process noise. Therefore, we need a new data structure to handle noise process directly.

In this thesis, using the Markov chain approximation method [43] and the rapidly-exploring sampling technique [51], we introduce a novel sampling-based algorithm called the *incremental Markov Decision Process* (iMDP) to approximately solve a wide class of stochastic optimal control problems. Unlike exploring trees in RRT-like algorithms, the iMDP algorithm uses a sequence of Markov Decision Processes to address the difficulty caused by process noise. The details of the iMDP algorithm will be presented in Chapter 3.

1.3 Risk Management

Risk management in stochastic optimal control has also received extensive attention by researchers in several fields. Broadly speaking, risk can be defined as a situation involving exposure to danger. In practice, we are often concerned with *several* additional requirements of control policies when minimizing an objective function. For example, trajectory performance requirements such as fuel consumption requirements on autonomous cars, stealthiness requirements for aircraft, thermal control requirements on spacecraft (e.g., to avoid long exposure of radiators to the Sun), and bounded collision probability are critical and must be respected while minimizing the time to execute a task. Controlled systems are considered to be in risky situations when these requirements are not met. Thus, we refer to theserequirements as *risk constraints.*

In this thesis, we consider risk constraints that are expressed as either bounded trajectory performance, which has the same structure as the objective function, or bounded probability of failure. The mathematical formulation of these constraints will be presented in Chapters 4 and 5 respectively. In the following, we briefly review the literature of constrained stochastic optimal control problems from multiple research communities.

Bounded trajectory performance

The management science community has focused on bounded trajectory performance constraints for discrete-time, finite-state MDP problems that arise from new technology management and production management. The considered bounded trajectory performance constraints also have the same structure as the objective function with possibly different discount factors. In [107,108], Feinberg and Shwartz consider these problems when constraints are applied *for particular initial states.* Thus, optimal control policies depend on the initial state. For this class of problems, the authors characterize optimal policies as a class of *nonstationary randomized policies.* In particular, if a feasible policy exists, then there exists an optimal policy which is stationary deterministic from some steps onward and randomized Markov before this step, but the number of randomized decisions is bounded by the number of constraints. The authors further argue that this class of nonstationary randomized policies is the simplest optimal policies for constrained stochastic optimal control problems with *different discount factors.*

A mixed linear-integer programming is also proposed to find this class of optimal policies [107, 108]. Thus, a possible method to solve continuous-time continuous-space stochastic optimal control in the presence of bounded trajectory performance constraints is to discretize these problems in both time and space. However, due to a large number of states and a large number of integer variables, this approach presents enormous computational challenges.

In this work, we enforce bounded trajectory performance constraints *for all subtrajectories.* This formulation imposes a stronger requirement for control policies and allow us to extend the iMDP algorithm to find *anytime stationary deterministic policies*, which are suitable for practical applications. The details of the extended iMDP algorithm for this class of constrained stochastic control problems are presented in Chapter 4.

Bounded probabilities of failure

In robotics, a common risk management problem is formulated as *chance-constrained optimization* [75, 109–113]. Historically, chance constraints specify that starting from *a given initial state*, the *time-0* probability of success must be above a given threshold where success means reaching goal areas safely. Alternatively, we call these constraints risk constraints (as done in this thesis) if we concern more about failure probabilities. For critical applications such as self-driving cars and robotic surgery, regulatory authorities can impose a threshold of failure probability during operation of these systems. Thus, finding control policies that fully respect this type of constraint is important in practice.

Despite intensive work done to solve this problem over the last 20 years, designing computationally efficient algorithms that respect chance constraints for systems with continuous-time dynamics is still an open question. The Lagrangian approach [32, 114, 115] is a possible method for solving the mentioned constrained optimization. However, this approach requires numerical procedures to compute Lagrange multipliers before obtaining a policy, which is often computationally demanding for high dimensional systems.

In another approach (see, e.g., [75, 112, 113, 116, 117]), most previous works use discrete-time multi-stage formulations to model this problem. In these modified formulations, failure is defined as collision with convex obstacles which can be represented as a set of linear inequalities. Probabilities of safety for states at different time instants as well as for the entire path are pre-specified by users. The proposed algorithms to solve these formulations often involve two main steps. In the first step, these algorithms often use heuristic [116] or iterative [117] *risk allocation* procedures to identify the tightness of different constraints. In the second step, the formulations with identified active constraints can be solved using mixed integer-linear programming with possible assistance of particle sampling [109] and linear programming relaxation [110]. Computing risk allocation fully is computationally intensive. Thus, in more recent works [75, 112, 113], the authors make use of the RRT and RRT* algorithms to build tree data structures that also store incremental approximate allocated risks at tree nodes. Based on the RRT* algorithm, the authors have proposed the Chance-Constrained-RRT* (CC-RRT*) algorithm that would provide asymptotically-optimal and probabilistically-feasible trajectories for linear Gaussian systems subject to process noise, localization error, and uncertain environmental constraints. In addition, the authors have also proposed a new objective function that allows users to trade-off between minimizing path duration and risk-averse behavior by adjusting the weights of these additive components in the objective function.

We note that the modified formulations in the above approach do not preserve well the intended guarantees of the original chance constraint formulation that specifies the bounded probability of failure from time-0 for only a particular initial state. In addition, although the recent developed algorithms can provide asymptotically-optimal and probabilistically-feasible trajectories, the approach requires the direct representation of convex obstacles into the formulations, which limits its use in practice. Solving the resulting mixed integer-linear programming when there is a large number of obstacles is computationally demanding. The proposed algorithms are also over-conservative due to loose union bounds when performing the risk allocation procedures. To counter these conservative bounds, CC-RRT* constructs more aggressive trajectories by adjusting the weights of the path duration and risk-averse components in the objective function. As a result, it is hard to automate the selection of trajectory patterns.

Moreover, specifying in advance probabilities of safety for states at different time instants and for the entire path can lead to policies that have irrational behaviors due inconsistent risk preference over time. This phenomenon is known as *time-inconsistency* of control policies. For example, when we execute a control policy returned by one of the proposed algorithms, due to noise, the system can be in an area surrounded by obstacles at some later time t, it would be safer if the controller takes into account this situation and increases the required probability of safety at time t to encourage careful maneuvers. Similarly, if the system enters an obstacle-free area, the controller can reduce the required probability of safety at time t to encourage more aggressive maneuvers. Therefore, to maintain time-consistency of control policies, the controller should adjust safety probabilities that are contingent on available information along the controlled trajectory.

In other related works [119–121], several authors have proposed new formulations in which the objective functions and constraints are evaluated using (different) single-period risk metrics. However, these formulations again lead to potential inconsistent behaviors as risk preferences change in an irrational manner between periods [122]. Recently, in [111], the authors used Markov dynamic time-consistent risk measures [123–125] to assess the risk of future cost stream in a consistent manner and established a dynamic programming equation for this modified formulation. The resulting dynamic programming equation has functionals over the state space as control variables. When the state space is continuous, the control space has inifinite dimensionality, and therefore, solving the dynamic programming equation in this case is computational challenging.

In mathematical finance, closely-related problems have been studied in the context of hedging with portfolio constraints where constraints on terminal states are enforced almost surely (a.s), yielding so-called *stochastic target problems* [21–25]. Research in this field focuses on deriving HJB equations for this class of problems. Recent analytical tools such as weak dynamic programming [21] and geometric dynamic programming [126, 127] have been developed to achieve this goal. These tools allow us to derive HJB equations and find viscosity solutions for a larger class of problems while avoiding measurability issues.

In this thesis, we consider the above stochastic optimal control problems with risk constraints that are expressed in terms of time-0 bounded probabilities of failure for *particular initial states.* As we will show in Chapter 5, we present a martingale approach to solve these problems such that obtained control policies are time-consistent with the initial threshold of failure probability. The martingale approach enables us to transform a risk-constrained problem into a stochastic target problem. The martingale represents the consistent variation of risk tolerance that is contingent on available information over time. The iMDP algorithm is then extended to compute anytime policies for the original constrained problem. It turns out that returned policies by the extended iMDP algorithm belong to a class of randomized policies in the original control policy space.

1.4 Statement of Contributions

The main contribution of this thesis is the development of theoretical foundations, and provably-correct and efficient sampling-based algorithms to solve continuous-time, continuous-space stochastic optimal control problems in the presence of complex risk constraints.

More specifically, the contributions of this thesis are listed as follows. In the first part of the thesis, we consider the mentioned problems without risk constraints. We propose a novel algorithm called the incremental Markov Decision Process (iMDP) to compute incrementally *any-time* control policies that approximate arbitrarily well an optimal policy in terms of the expected cost.

The main idea is to generate an approximating data structure which is a sequence of finite discretizations of the original problem through random sampling of the state space. At each iteration, the discretized problem is a Markov Decision Process that serves as an incrementally refined model of the original problem. That is, the discrete MDP is refined by adding new states sampled from the boundary as well as from the interior of the state space. Subsequently, new stochastic transitions are constructed to connect the new states to those already in the model. For the sake of efficiency, stochastic transitions are computed only when needed. Then, an anytime policy for the refined model is computed using an incremental value iteration algorithm, based on the value function of the previous model. This process is iterated until convergence. The policy for the discrete system is finally converted to a policy for the original continuous problem.

With probability one, we show that:

- The sequence of the optimal value functions for each of the discretized problems converges uniformly to the optimal value function of the original stochastic optimal control problem, and
- The original optimal value function can be computed efficiently in an incremental manner using asynchronous value iterations.

Thus, the proposed algorithm provides an anytime approach to the computation of optimal control policies of the continuous problem. In fact, the distributions of approximating trajectories and control processes returned by the iMDP algorithm approximate arbitrarily well the distributions of optimal trajectories and optimal control processes of the continuous problem.

Moreover, each iteration of the iMDP algorithm can be implemented with the time complexity $O(n^\theta(\log n)^2)$ per iteration where the parameter θ belongs to $(0, 1]$, and n is the number of states in an MDP model in the algorithm which increases linearly due to our sampling strategy. Therefore, the iMDP algorithm guarantees asymptotic optimality while maintaining low computational and space complexity. Compared to the time complexity per iteration $O(\log n)$ of RRT and RRT*, the complexity of iMDP algorithm is slighly higher in order to handle uncertainty and provide closed-loop control policies.

The iMDP algorithm provides several benefits for solving stochastic optimal control problems:

- The iMDP algorithm is an *algorithmic method* to construct approximate solutions without the need to derive and characterize viscosity solutions of the associated HJB equations. Hence, the algorithm is suitable for a very broad class of stochastic control problems where HJB equations are not well understood.

- The underlying probabilistic convergence proof of the Markov chain approximation method holds true even for complex stochastic dynamics with discontinuity and jumps. Thus, the iMDP algorithm is capable of handling such complex system dynamics.

- As the approximating MDP sequence is constructed incrementally using a collision-checking test, the iMDP is particularly suitable for online robotics applications without *a priori* discretization of the state space in cluttered environments.

- The iMDP algorithm also has an important *anytime* flavor in its computation. The algorithm tends to provide a feasible solution quickly, and when additional computation time is available, the algorithm continues refining the solution.

In the second part of the thesis, we consider risk constraints that are expressed as either bounded trajectory performance or bounded probabilities of failure. For bounded trajectory performance constraints, we enforce these constraints for all sub-trajectories. We extend the iMDP algorithm to approximate arbitrarily well an optimal feedback policy of the constrained problem. We show that the sequence of policies returned from the extended algorithm are both probabilistically sound and asymptotically optimal.

For bounded failure probability constraints enforced for particular initial states, we present a martingale approach that diffuses a risk constraint into a martingale to construct time-consistent control policies. The martingale stands for the level of risk tolerance over time. By augmenting the system dynamics with the martingale, the original risk-constrained problem is transformed into a stochastic target problem. We extend the iMDP algorithm to approximate arbitrarily well an optimal feedback policy of the original problem by sampling in the augmented state space and computing proper boundary values for the reformulated problem. We also show that the sequence of policies returned from the extended algorithms are both probabilistically sound and asymptotically optimal in the original control policy space. Furthermore, anytime control policies in this case are randomized policies.

The effectiveness of the iMDP algorithm and its extended versions is demonstrated on robot motion planning and control problems in cluttered environments in the presence of process noise.

Lastly, the final chapter of the thesis points out several important directions for future research such as parallel and distributed implementation of iMDP algorithms, stochastic control with logic constraints, novel sampling-based methods to handle sensor information, and stochastic differential games. The ultimate goal of this research direction is to achieve high degree of autonomy for systems to operate safely in uncertain and highly dynamic environments with complex mission specifications.

1.5 Outline

This thesis is organized as follows:

- In Chapter 2, we will present preliminary concepts, mathematical definitions and notations for our discussion in the following chapters. We will introduce several models for continuous-time stochastic system dynamics and approximating discrete structures for the continuous dynamics. Well-known results for these models will be presented for future reference in later chapters.

- In Chapter 3, we will formulate the standard continuous-time continuous-space stochastic optimal control problem. The incremental Markov Decision Process (iMDP) algorithm will be presented to provide asymptotically-optimal solutions using efficient incremental computation. We will also provide detailed analysis of the iMDP algorithm and present several experimental results to support the analysis.

- In Chapter 4, we will present a class of stochastic optimal control in the presence of bounded trajectory performance constraints. This is the first type of risk constraints that we consider in this thesis. We extend the iMDP algorithm to provide probabilistically-sound and asymptotically-optimal policies in an anytime manner for this class of constrained problems.

- In Chapter 5, we will consider stochastic optimal control problems subject to the second type of risk constraints that are formulated as bounded probabilities of failure. We will introduce a martingale approach to convert these probability constraints into controlled martingales so that we would instead solve equivalent stochastic target problems. As a result, we can extend the iMDP algorithm to provide probabilistically-sound and asymptotically-optimal policies to the transformed problems. We then convert these policies into anytime policies of the original constrained problems.

- Finally, in Chapter 6, we conclude the thesis and present future research directions.

Chapter 2

Background and Preliminaries

In this chapter, we first present formal notations and definitions used in thesis. We then overview important results that lay the foundations to analyze the algorithms presented in the following chapters. In particular, we will review Brownian motion, controlled diffusion processes, and random geometric graphs. During our discussion in the next chapters, we will remind these notations, definitions, and results when necessary. The details of these materials can be found in [43, 104, 128, 129].

2.1 Basic Definitions and Notations

Convergence

We denote $\mathbb{N}$ as the set of natural numbers starting from 1, $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$, and $\mathbb{R}$ as the set of real numbers. Similarly, $\mathbb{R}^k$ is the set of k-dimensional real vectors. We also denote $\overline{\mathbb{R}}$ as the set of extended real numbers, i.e. $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$. A sequence on a set X is a mapping from $\mathbb{N}_0$ to X, denoted as $\{x_n\}_{n=0}^{\infty}$, where $x_n \in X$ for each $n \in \mathbb{N}$. Given a metric space X endowed with a metric d, a sequence $\{x_n\}_{n=0}^{\infty} \subset X$ is said to converge if there is a point $x \in X$, denoted as $\lim_{n\to\infty} x_n$, with the following property: For every $\epsilon > 0$, there is an integer N such that $n \geq N$ implies that $d(x_n, x) < \epsilon$.

A sequence of functions $\{f_n\}_{n=1}^{\infty}$ in which each function f_n is a mapping from X to $\mathbb{R}$ *converges pointwise* to a function f on X if for every $x \in X$, the sequence of numbers $\{f_n(x)\}_{n=0}^{\infty}$ converges to $f(x)$. A sequence of functions $\{f_n\}_{n=1}^{\infty}$ *converges uniformly* to a function f on X if the following sequence $\{M_n \mid M_n = \sup_{x\in X} |f_n(x) - f(x)|\}_{n=0}^{\infty}$ converges to 0.

Measurable space

Let X be a set. A *σ-algebra* $\mathcal{A}$ on a set X is a collection of subsets of X that contains the empty set, the set X itself, and is closed under complement and countable union of its members. The tuple $(X, \mathcal{A})$ is called a *measurable space.* Let $(X, \mathcal{A})$ and $(Y, \mathcal{B})$ be measurable spaces. A function $f : X \to Y$ is an *$\mathcal{A}$-measurable function* if $f^{-1}(B) \in \mathcal{A}$ for every $B \in \mathcal{B}$. A σ-algebra generated by the function f is defined as

$\sigma(f) = \{f^{-1}(B) \mid B \in \mathcal{B}\}$. Thus, f is $\mathcal{A}$-measurable if $\sigma(f) \subset \mathcal{A}$. When it is clear from the context, we use $\mathcal{A}$-measurable and measurable interchangeably.

A Borel set is any set in a topological space that can be formed from open sets through the operations of countable union, countable intersection, and complement. If S is a topological space, we denote by $\mathcal{B}(S)$ the σ-algebra of Borel subsets of S.

Probability space

Let us consider a random experiment $\mathcal{E}$. The sample space Ω of $\mathcal{E}$ is a set of all possible outcomes ω of $\mathcal{E}$. Let $\mathcal{F}$ be *a* σ-algebra on Ω such that $(\Omega, \mathcal{F})$ is a measurable space, then $\mathcal{F}$ is an event space of $\mathcal{E}$. A subset A of $\mathcal{F}$ is called an event. The complement of an event A is denoted as A^c. A *probability measure* P is a mapping from $\mathcal{F}$ to $\mathbb{R}$ satisfying the following three axioms: (i) the probability $P(A)$ of an event $A \in \mathcal{F}$ occurring is a real number between 0 and 1, (ii) the probability $P(\Omega)$ of the event Ω occurring is 1, and (iii) the probability of any of countably many pairwise disjoint events ocurring is the sum of the probabilities of the occurrence of each of the individual events. The tuple $(\Omega, \mathcal{F}, P)$ is called a *probability space* of the experiment $\mathcal{E}$.

Two events A, B are *independent* if $P(A \cap B) = P(A)P(B)$. Two σ-algebras $\mathcal{A}, \mathcal{B} \subset \mathcal{F}$ are independent if for any $A \in \mathcal{A}$ and $B \in \mathcal{B}$, A and B are independent. A *random variable* is a measurable function mapping from Ω to $\mathbb{R}$.

The construction of a probability space can be incremental in the following sense. We say that a probability $(\Omega', \mathcal{F}', P')$ extends another probability space $(\Omega, \mathcal{F}, P)$ if there exists a surjective map $\pi : \Omega' \to \Omega$ which is measurable, i.e., $\pi^{-1}(A) \in \mathcal{F}'$ for every $A \in \mathcal{F}$, and probability preserving, i.e., $P'(\pi^{-1}(A)) = P(A)$ for every $A \in \mathcal{F}$ [130]. An event A in the original probability space is canonically identified with an event $\pi^{-1}(A)$ in the extended probability space. Thus, insead of specifying in advance a probability space having a rich enough structure so that all random variables of interest can be defined, we can extend a probability space when necessary to define new random variables. This is a useful probabilistic way of thinking, especially when we study stochastic processes, so that the sample space Ω can be considered as an ambient sample space.

Convergence of random variables

Let us consider a probability space $(\Omega, \mathcal{F}, P)$. Given a sequence of events $\{A_n\}_{n=0}^{\infty}$, we define $\limsup_{n\to\infty} A_n$ as $\cap_{n=0}^{\infty} \cup_{k=n}^{\infty} A_k$, i.e., the event that A_n occurs infinitely often. In addition, the event $\liminf_{n\to\infty} A_n$ is defined as $\cup_{n=0}^{\infty} \cap_{k=n}^{\infty} A_k$. The expected value of a random variable Y is defined as $\mathbb{E}[Y] = \int_{\Omega} Y dP$ using the *Lebesgue integral.*

A sequence of random variables $\{Y_n\}_{n=0}^{\infty}$ *converges surely* to a random variable Y if $\lim_{n\to\infty} Y_n(\omega) = Y(\omega)$ for all $\omega \in \Omega$. A sequence of random variables $\{Y_n\}_{n=0}^{\infty}$ *converges almost surely* or *with probability one* (w.p.1) to a random variable Y if $P(\omega \in \Omega \mid \lim_{n\to\infty} Y_n(\omega) = Y(\omega)) = 1$. Almost sure convergence of $\{Y_n\}_{n=0}^{\infty}$ to Y is denoted as $Y_n \overset{a.s.}{\to} Y$.

We say that a sequence of random variables $\{Y_n\}_{n=0}^{\infty}$ *converges in probability* to a random variable Y, denoted as $Y_n \xrightarrow{p} Y$ or $\text{plim}_{n\to\infty} Y_n = Y$, if for every $\epsilon > 0$, we have $\lim_{n\to\infty} P(|X_n - X| \geq \epsilon) = 0$. For every continuous function $f(\cdot)$, if $Y_n \xrightarrow{p} Y$, then we also have $f(Y_n) \xrightarrow{p} f(Y)$. Moreover, if $Y_n \xrightarrow{p} Y$ and $Z_n \xrightarrow{p} Z$, then $(Y_n, Z_n) \xrightarrow{p} (Y, Z)$.

We say that a sequence of random variables $\{Y_n\}_{n=0}^{\infty}$ *converges in distribution* to a random variable Y if $\lim_{n\to\infty} F_n(x) = F(x)$ for every $x \in \mathbb{R}$ at which F is continuous where $\{F_n\}_{n=0}^{\infty}$ and F are the associated CDFs of $\{Y_n\}_{n=0}^{\infty}$ and Y respectively. We denote this convergence as $Y_n \xrightarrow{d} Y$. Convergence in distribution is also called weak convergence. If $Y_n \xrightarrow{d} Y$, then $\lim_{n\to\infty} \mathbb{E}[f(Y_n)] = \mathbb{E}[f(Y)]$ for all bounded continuous functions f. As a corollary, when $\{Y_n\}_{n=0}^{\infty}$ converges in distribution to 0, and Y_n is bounded for all n, we have $\lim_{n\to\infty} \mathbb{E}[Y_n] = 0$ and $\lim_{n\to\infty} \mathbb{E}[Y_n^2] = 0$, which together imply $\lim_{n\to\infty} \text{Var}(Y_n) = 0$. We also have if $|Z_n - Y_n| \xrightarrow{p} 0$ and $Y_n \xrightarrow{d} Y$, we have $Z_n \xrightarrow{d} Y$.

In addition, an event $E(n)$, which depends on a parameter n, holds *asymptotically almost surely* if $\lim_{n\to\infty} P(E(n)) = 1$. Thus, when $Y_n \xrightarrow{p} Y$, then this implies that the event $Y_n = Y$ happens asymptotically almost surely, i.e. $\lim_{n\to\infty} P(Y_n = Y) = 1$.

Finally, we say that a sequence of random variables $\{Y_n\}_{n=0}^{\infty}$ *converges in r^{th} mean* to a random variable Y, denoted as $Y_n \xrightarrow{r} Y$, if $\mathbb{E}[|X_n|^r] < \infty$ for all n, and $\lim_{n\to\infty} \mathbb{E}[|X_n - X|^r] = 0$.

We have the following implications: (i) almost sure convergence or r^{th} mean convergence ($r \geq 1$) implies convergence in probability, and (ii) convergence in probability implies convergence in distribution. The above results still hold for random vectors in higher dimensional spaces.

Conditional expectation

On a probability space $(\Omega, \mathcal{F}, P)$, let $A \in \mathcal{F}$ be an event such that $P(A) > 0$. The conditional probability of an event B given the event A, denoted as $P(B \mid A)$, is defined as $P(B \mid A) = P(B \cap A)/P(A)$. Let $Q : \mathcal{F} \to \mathbb{R}$ such that $Q(B) = P(B \mid A)$ then Q is a probability measure on $(\Omega, \mathcal{F})$. Conditional expectation of a random variable X given the event A is defined as $\mathbb{E}[X \mid A] = \int_\Omega X dQ$ whenever this integration is well defined. Let $1_A : \omega \to \{0, 1\}$ be an indicator function that takes value 1 if $\omega \in A$ and 0 otherwise. When $\mathbb{E}[|X|1_A] < \infty$, then X is Q-integrable and $\mathbb{E}[X \mid A] = \mathbb{E}[X1_A]/P(A)$.

Conditional expectation can also be defined with respect to a σ-algebra and a random variable. Let $\mathcal{G} \subset \mathcal{F}$ be a sub σ-algebra, the conditional expectation of a random variable X given $\mathcal{G}$, $\mathbb{E}[X \mid \mathcal{G}]$, is the unique $\mathcal{G}$-measurable random variable Z such that $\mathbb{E}[X1_G] = \mathbb{E}[Z1_G]$ for all $G \in \mathcal{G}$. Furthermore, a conditional expectation of a random variable X given a random variable Y is defined as $\mathbb{E}[X \mid Y] = \mathbb{E}[X \mid \sigma(Y)]$.

Conditional expectation has following properties. For any two random variable X, Y, and $\alpha, \beta \in \mathbb{R}$, we have $\mathbb{E}[\alpha X + \beta Y \mid \mathcal{G}] = \alpha\mathbb{E}[X \mid \mathcal{G}] + \beta\mathbb{E}[Y \mid \mathcal{G}]$. For any random variable X, we have $\mathbb{E}[\mathbb{E}[X \mid \mathcal{G}]] = \mathbb{E}[X]$. When X is $\mathcal{G}$-measurable, $\mathbb{E}[X \mid \mathcal{G}] = X$. When X and $\mathcal{G}$ are independent, which means $\sigma(X)$ and $\mathcal{G}$ are independent, then $\mathbb{E}[X \mid \mathcal{G}] = \mathbb{E}[X]$. If $\mathcal{G}_1 \subset \mathcal{G}_2 \subset \mathcal{F}$, then $\mathbb{E}[\mathbb{E}[X \mid \mathcal{G}_1] \mid \mathcal{G}_2] = \mathbb{E}[\mathbb{E}[X \mid \mathcal{G}_2] \mid \mathcal{G}_1] = \mathbb{E}[X \mid \mathcal{G}_1]$. When Y and XY are integrable, X is $\mathcal{G}$-measurable, then $\mathbb{E}[XY \mid \mathcal{G}] = X\mathbb{E}[Y \mid \mathcal{G}]$ a.s. If $X \leq Y$ a.s., then $\mathbb{E}[X \mid \mathcal{G}] \leq \mathbb{E}[Y \mid \mathcal{G}]$ a.s.

Stochastic processes

A *stochastic process* is a *collection of random variables* indexed by time. That is, consider any indexing set $I \subset \mathbb{R}$, we call $\{X(t); t \in I\}$ a stochastic process on a probability space $(\Omega, \mathcal{F}, P)$ when each $X(t)$ is a random variable for all $t \in I$. When $I = \mathbb{N}$, $\{X(t); t \in \mathbb{N}\}$ a discrete-time stochastic process. When $I = [0, \infty)$, $\{X(t); t \geq 0\}$ is a continuous-time stochastic process. Thus, X is a mapping from $I \times \Omega$ to $\mathbb{R}$, and $X(t, \omega)$ is the value of the process at time t for an outcome ω. Fixing ω, $X(\cdot, \omega)$ is called a *sample path* for ω. From this perspective, a *stochastic process* is a *collection of sample paths* $\{X(\cdot, \omega) : \omega \in \Omega\}$. We can suppress ω and refer to the stochastic process as $X(\cdot)$.

The following notations are handy to refer to several special classes of sample paths. Let $C^k[0, T]$ denote the space of continuous sample path functions mapping from $[0, T]$ to $\mathbb{R}^k$, and $D^k[0, T]$ denote the space of those functions from $[0, T]$ to $\mathbb{R}^k$ that are continuous from the right and have limits from the left. Let $C^k[0, \infty)$ and $D^k[0, \infty)$ denote the analogous path spaces on the interval $[0, \infty)$ respectively. Given an open set U on some Euclidean space, let $C^k(U)$ be the set of all real-valued functions on U that have continuous derivatives up to and including order k.

Filtrations and martingale

In probability theory, filtrations are used to model the sequence of refined information available over time. Let us consider a probability space $(\Omega, \mathcal{F}, P)$. A family of σ-algebra $\{\mathcal{F}_t; t \geq 0\}$ is called *a filtration* on this probability space if $\mathcal{F}_s \subset \mathcal{F}_t \subset \mathcal{F}$ for all $0 \leq s \leq t$. Intuitively, $\mathcal{F}_t$ is the collection of events whose occurrence can be determined up to time t. An $\mathcal{F}_t$-measurable random variable is one whose value can be *determined by time t*. If X is any random variable, $\mathbb{E}[X \mid \mathcal{F}_t]$ is the "best" estimate of X (in the sense of least mean square errors) based on information up to time t.

A process $\{X(t); t \geq 0\}$, or simply $M(\cdot)$, is $\mathcal{F}_t$-*adapted* to the filtration $\{\mathcal{F}_t; t \geq 0\}$ if the random variable $X(t)$ is $\mathcal{F}_t$-measurable (i.e. its value is known at time t). We say that a process $M(\cdot)$ is an $\mathcal{F}_t$-*martingale* if $M(\cdot)$ is an $\mathcal{F}_t$-adapted process such that $\mathbb{E}[|M(t)|] < \infty$ for all $t \geq 0$ and $\mathbb{E}[M(t+s) \mid \mathcal{F}_t] = M(t)$ for all $s, t > 0$ (i.e. the current value is the best estimate for future values).

A random variable $\tau : \Omega \to [0, \infty]$ is called an $\mathcal{F}_t$-*stopping time* if the event $\{\tau \leq t\} \in \mathcal{F}_t$ for all $t \in [0, \infty]$. If $M(\cdot)$ is an $\mathcal{F}_t$-martingale and τ is a uniformly bounded $\mathcal{F}_t$-stopping time, the stopped process $M(t \wedge \tau)$ is also an $\mathcal{F}_t$-martingale where $t \wedge \tau$ is the minimum of t and τ. Again, when the particular filtration is obvious, we will suppress the prefix and refer to $M(\cdot)$ and τ as a martingale and a stopping time.

Complexity

Let $f(n)$ and $g(n)$ be two functions with domain and range $\mathbb{N}$ or $\mathbb{R}$. The function $f(n)$ is called $O(g(n))$ if there exists two constants M and n_0 such that $f(n) \leq Mg(n)$ for all $n \geq n_0$. The function $f(n)$ is called $\Omega(g(n))$ if $g(n)$ is $O(f(n))$. Finally, the function $f(n)$ is called $\Theta(g(n))$ if $f(n)$ is both $O(g(n))$ and $\Omega(g(n))$.

2.2 Probabilistic Models

In this section, we introduce Brownian motion and review controlled diffusion processes that are used to model system dynamics in this work. We then present classical results on the existence and uniqueness of controlled processes in this model.

2.2.1 Brownian motion

Definition 2.2.1 (Brownian motion) *Let $(\Omega, \mathcal{F}, P)$ be a probability space and $\{\mathcal{F}_t; t \geq 0\}$ be a filtration defined on it. A process $\{w(t); t \geq 0\}$ is called an $\mathcal{F}_t$-Wiener process or an $\mathcal{F}_t$-Brownian motion if it satisfies the following conditions:*

a. $w(0) = 0$ w.p.1.

b. $w(t)$ is $\mathcal{F}_t$-measurable, and $\sigma(w(s) - w(t))$ is independent of $\mathcal{F}_t$ for all $s \geq t \geq 0$.

c. $w(t) - w(s)$ is a Normal random variable, $N(0, \varphi^2(s-t))$, for all $s > t \geq 0$.

d. The sample paths of $w(\cdot)$ are continuous real-valued functions in $C[0, \infty)$.

When $\varphi = 1$, the process is called a standard Brownian motion.

The constructions of an $\mathcal{F}_t$-Brownian motion are described in the book of Karatzas and Shreve [128]. When the filtration $\{\mathcal{F}_t; t \geq 0\}$ is actually generated by $w(\cdot)$, i.e. $\mathcal{F}_t = \sigma(w(s) : 0 \leq s \leq t)$, the prefix $\mathcal{F}_t$ can be suppressed. In such case, $\mathcal{F}_t$ is the collection of events whose occurrence can be determined from observations of the Brownian motion $w(\cdot)$ by time t.

Brownian motion defines a probability measure on the space $C[0, \infty)$ of continuous sample paths, called Wiener measure. Formally, a Wiener measure is a mapping from a σ-algebra $\overline{\mathcal{F}}$ on $C[0, \infty)$ to $[0, 1]$ and can be constructed using Carathéodory's theorem [128].

In the following discussion, if otherwise noted, we will always consider standard Brownian motions. Although Brownian sample paths are not differentiable pointwise, we can interpret their derivative in a distributional sense as follows.

Definition 2.2.2 (Differential of Brownian motion) *The differential $dw(t)$ of a standard Brownian motion is the following limit:*

$$dw(t) = \lim_{\Delta t \to dt} \left(w(t + \Delta t) - w(t) \right) \tag{2.1}$$

Since $w(t+\Delta t) - w(t)$ is $N(0, \Delta t)$, after passing the limit, we have $dw(t)$ is $N(0, dt)$. For this reason, we have the following identity:

$$\big(dw(t)\big)^2 = dt. \tag{2.2}$$

It also follows that $dt.dw(t) = o(dt)$, and $dt.dt = o(dt)$. We recognize that $dw(t)/dt$ is $N(0, 1/dt)$ with infinite variance when $dt \to 0$. In engineering, we refer to the stochastic process $v(t) = dw(t)/dt$ as *white noise.*

Finally, Brownian motion can also be defined in multi-dimensional spaces:

Definition 2.2.3 (Multi-dimensional Brownian motion) *An m-dimensional $\mathcal{F}_t$-Brownian motion $w(\cdot)$ is a process $\big(w_1(\cdot), ..., w_m(\cdot)\big)$ taking values in $\mathbb{R}^m$ in which $\{w_j(\cdot)\}_{j=1}^m$ are m independent $\mathcal{F}_t$-Brownian motions.*

Thus, when $w(t)$ is standard, $w(t) - w(s)$ is a multivariate Normal random variable, $N\big(0, (s-t)I_{m\times m}\big)$, for all $s > t \geq 0$ where $I_{m\times m}$ is an m by m identity matrix.

Stochastic integration

Let us consider a probability space $(\Omega, \mathcal{F}, P)$, and $\{\mathcal{F}_t; t \geq 0\}$ is a filtration on this space. A continuous-time stochastic process $X(\cdot)$ is called *measurable* if $\{(t,\omega) : X(t,\omega) \in A\}$ belongs to the product σ-algebra $\mathcal{B}([0,\infty)) \times \mathcal{F}$ for any $A \subset \mathbb{R}$ where $\mathcal{B}([0,\infty))$ is the σ-algebra of Borel subsets of $[0,\infty)$. Let $\Sigma_b(T)$ denote the set of $\mathcal{F}_t$-adapted, measurable, real-valued processes $F(\cdot)$ which are uniformly bounded in $t \in [0,T]$ and $\omega \in \Omega$. Let Σ_b denote those processes defined on $[0,\infty)$ that are in $\Sigma_b(T)$ for each $T < \infty$.

Let $w(\cdot)$ be a standard $\mathcal{F}_t$-Brownian motion and let F be a stochastic process in Σ_b, the *Itô integration* of $F(\cdot)$ against $w(\cdot)$ up to time t is a stochastic process $Y(\cdot)$ denoted as:

$$Y(t) = \int_0^t F(\tau)dw(\tau).$$

The above integral is formally defined via simple functions in Σ_b. The details of this construction are in the book of Karatzas and Shreve [128].

Let $f(\cdot)$ be an $\mathcal{F}_t$-adapted Lebesgue-integrable stochastic process, we call a process $x(\cdot)$ an *Itô process* if its value evolves over time as follows:

$$x(t) = x(0) + \int_0^t f(\tau)d\tau + \int_0^t F(\tau)dw(\tau). \tag{2.3}$$

The above equation can be written equivalently as:

$$dx(t) = f(t)dt + F(t)dw(t). \tag{2.4}$$

Since $dw(t)/dt$ is interpreted as white noise, the above equation models the process $x(\cdot)$ with a drift component specified by $f(\cdot)$ and an additive noise magnified by $F(\cdot)$.

We often need to compute a stochastic process that is a function of an Itô process using the following lemma:

Lemma 2.2.4 (Itô lemma) *Let $w(\cdot)$ be a standard Brownian motion, and $x(\cdot)$ is a process satisfying $dx(t) = f(t)dt + F(t)dw(t)$ where f is adapted and Lebesgue-integrable, and $F \in \Sigma_b$. For any function $g \in C^2(\mathbb{R})$, we have:*

$$g(x(t)) = g(x(0)) + \int_0^t g_x(x(s))f(s)ds + \int_0^t g_x(x(s))F(s)dw(s) + \tfrac{1}{2}\int_0^t g_{xx}(x(s))F^2(s)ds, \quad (2.5)$$

which can be symbolically written as:

$$dg(x(t)) = \Big[g_x(x(t))f(t) + \frac{1}{2}g_{xx}(x(t))F^2(t)\Big]dt + g_x(x(t))F(t)dw(t), \qquad (2.6)$$

where g_x and g_{xx} are the first and second derivatives of g.

The above lemma extends naturally to higher dimensions. Let us consider an n-dimensional vector of adapted and Lebesgue-integrable process $f(\cdot)$, an $n \times m$ matrix Itô-integrable process $F(\cdot)$, an m-dimensional Brownian motion $w(\cdot)$, then $dx(t) = f(t)dt + F(t)dw(t)$ is an n-dimensional Itô process where:

$$dx_j(t) = f_j(t)dt + \sum_{k=1}^{m} F_{j,k}(t)dw_k(t).$$

For $g : \mathbb{R}^n \to \mathbb{R}^p$, the Itô lemma becomes:

$$dg(x(t)) = \left[\frac{\partial g^T(x(t))}{\partial x}f(t) + \frac{1}{2}Tr\left(F(t)F^T(t)\frac{\partial^2 g(x(t))}{(\partial x)^2}\right)\right]dt + \frac{\partial g^T(x(t))}{\partial x}F(t)dw(t),$$

where the gradient $\partial g/\partial x$ and Hessian $\partial^2 g/(\partial x)^2$ are evaluated at $x(t)$.

Importantly, we have the following well-known martingale representation theorem to relate a martingale and an Itô integral.

Theorem 2.2.5 (Martingale representation theorem) *Suppose $M(\cdot)$ is an $\mathcal{F}_t$-martingale where $\{\mathcal{F}_t; t \geq 0\}$ is the filtration generated by the m-dimensional standard Brownian motion $w(\cdot)$. If $\mathbb{E}[M(t)^2] < \infty$ for all t, then there exists a unique m-dimensional adapted stochastic process, $\phi(\cdot)$ such that*

$$M(t) = M(0) + \int_0^t \phi_s^T dw(t).$$

That is, every martingale is an initial condition plus an Itô integral with respect to the driving Brownian motion.

2.2.2 Controlled diffusion processes

When $f(\cdot)$ and $F(\cdot)$ are functions of the stochastic process $x(\cdot)$, we have a *stochastic differential equation* (SDE):

$$dx(t) = f(x(t))dt + F(x(t))dw(t).$$

In this work, we further consider those $f(\cdot)$ and $F(\cdot)$ functions that depend on external control variables in stochastic optimal control problems. Such models are called controlled diffusion processes.

Formally, let $(\Omega, \mathcal{F}, P)$ be a probability space. On this probability space, we define a filtration $\{\mathcal{F}_t; t \geq 0\}$ and an d_w-dimensional $\mathcal{F}_t$-Brownian motion. Let U be a compact subset of $\mathbb{R}^{d_u}$, and let $u(\cdot)$ be a U-valued, measurable process also defined on the same probability space. The control process $u(\cdot)$ is called *non-anticipative* with respect to the Brownian process $w(\cdot)$ if $u(\cdot)$ is also $\mathcal{F}_t$-adapted. In such case, we say $u(\cdot)$ is *admissible* with respect to $w(\cdot)$ or the pair $(u(\cdot), w(\cdot))$ is admissible. Let S be a bounded subset of $\mathbb{R}^n$, and let $f : S \times U \rightarrow \mathbb{R}^{d_x}$ and $F : S \times U \rightarrow \mathbb{R}^{d_x \times d_w}$ are bounded measurable and continuous functions. We consider *a controlled diffusion process* $x(\cdot)$ of the form:

$$dx(t) = f(x(t), u(t))dt + F(x(t), u(t))dw(t). \tag{2.7}$$

Given a control process $u(\cdot)$, a solution $x(\cdot)$ that solves Eq. (2.7) satisfies:

$$x(t) = x(0) + \int_0^t f(x(\tau), u(\tau))d\tau + \int_0^t F(x(\tau), u(\tau))dw(\tau). \tag{2.8}$$

In Eq. (2.7), $f(\cdot,\cdot)$ is called a drift vector, and $F(\cdot,\cdot)F^T(\cdot,\cdot)$ is called a diffusion matrix. We refer to $F(\cdot,\cdot)$ as a *dispersion matrix.* Roughly speaking, given $u(\cdot)$, $f(\cdot,\cdot)$, and $F(\cdot,\cdot)$, the process $x(\cdot)$ satisfies the following "local" properties for small time Δt:

$$\begin{aligned}\mathbb{E}[x(t+\Delta_t) - x(t) \mid x(t)] &\approx f(x(t), u(t))\Delta t,\\ \mathrm{Cov}[x(t+\Delta_t) - x(t) \mid x(t)] &\approx F(x(t), u(t))F^T(x(t), u(t))\Delta t.\end{aligned}$$

It turns out that these local properties are important and useful to construct consistent approximation of Eq. (2.7) as we will present in Chapter 3. In the following, we discuss different solution concepts, the existence and uniqueness of solutions to Eq. (2.7), and regularity conditions on f and F to have such solutions.

Definition 2.2.6 (Strong existence and uniqueness) *We say that strong existence of a solution holds for Eq.* (2.7) *if given a probability space* $(\Omega, \mathcal{F}, P)$*, a filtration* $\{\mathcal{F}_t; t \geq 0\}$*, an* $\mathcal{F}_t$*-Brownian motion* $w(\cdot)$*, an* $\mathcal{F}_t$*-adapted control process* $u(\cdot)$*, and an* $\mathcal{F}_0$*-measurable initial condition* $x(0)$*, then an* $\mathcal{F}_t$*-adapted process* $x(t)$ *exists satisfying Eq.* (2.8) *for all* $t > 0$.

Let $x_i(\cdot)$, $i = 1, 2$ that solve Eq. (2.7). *We say that strong uniqueness holds if*

$$P(x_1(0) = x_2(0)) = 1 \Rightarrow P(x_1(t) = x_2(t)\ \forall t > 0) = 1.$$

Definition 2.2.7 (Weak existence and uniqueness) *Let Γ be the sample path space of admissible pairs $(u(\cdot), w(\cdot))$. Suppose we are given probability distributions Λ and P_0 on Γ and on S respectively. We say that a solution of* (2.7) *exists in the weak sense if there exists a probability space $(\Omega, \mathcal{F}, P)$, a filtration $\{\mathcal{F}_t; t \geq 0\}$, an $\mathcal{F}_t$-Brownian motion $w(\cdot)$, an $\mathcal{F}_t$-adapted control process $u(\cdot)$, and an $\mathcal{F}_t$-adapted process $x(\cdot)$ satisfying Eq.* (2.8), *such that Λ and P_0 are the distributions of $(u(\cdot), w(\cdot))$ and $x(0)$ under P. We call such tuple $\{(\Omega, \mathcal{F}, P), \mathcal{F}_t, w(\cdot), u(\cdot), x(\cdot)\}$ a weak sense solution of Eq.* (2.7).

Assume that we are given weak sense solutions $\{(\Omega_i, \mathcal{F}_i, P_i), \mathcal{F}_{t,i}, w_i(\cdot), u_i(\cdot), x_i(\cdot)\}$, $i = 1, 2$ to Eq. (3.1). *We say solutions are weakly unique if equality of the joint distributions of $(w_i(\cdot), u_i(\cdot), x_i(0))$ under P_i, $i = 1, 2$, implies the equality of the distributions $(x_i(\cdot), w_i(\cdot), u_i(\cdot), x_i(0))$ under P_i, $i = 1, 2$.*

Intuitively, strong existence requires that the probability space, filtration, and Brownian motion are given *a priori*, and the solution then be found for the given data. Weak existence, on the other hand, allows these objects to be constructed together with the solution. Strong uniqueness is also called pathwise uniqueness, and weak uniqueness is also called uniqueness in the sense of probability distribution. Thus, strong existence and uniqueness imply weak existence and uniqueness. Moreover, weak existence and strong uniqueness together imply strong existence [128].

Several works have investigated regularity conditions for drift vectors and dispersion matrices to guarantee the existence and uniqueness of strong and weak solutions [128, 129]. In particular, the following results are useful in this thesis.

Theorem 2.2.8 (Conditions for strong uniqueness, see Theorem 5.2.5 in [128]) *Let us consider functions $f(\cdot, \cdot)$ and $F(\cdot, \cdot)$ that are locally Lipschitz-continuous in the space variable, i.e., for every integer $n \geq 1$, there exists $K_n \in (0, \infty)$ such that*

$$||f(x, u) - f(y, u)|| + ||F(x, u) - F(y, u)|| \leq K_n ||x - y||$$

for all $||x|| \leq n, ||y|| \leq n$, and $u \in U$. Then strong uniqueness holds for Eq. (2.7).

We require a stronger condition so that strong existence holds.

Theorem 2.2.9 (Conditions for strong existence, see Theorem 3.1 in [43] and Theorem 5.2.9 in [128]) *Let us consider functions $f(\cdot, \cdot)$ and $F(\cdot, \cdot)$ that are globally Lipschitz-continuous in the space variable, i.e. there exists $K \in (0, \infty)$ such that*

$$||f(x, u) - f(y, u)|| + ||F(x, u) - F(y, u)|| \leq K||x - y||$$

for all $x, y \in S$, and $u \in U$. Then for every deterministic initial condition $x(0)$, Eq. (2.7) *has a strong solution $x(\cdot)$.*

Furthermore, if $f(\cdot,\cdot)$ and $F(\cdot,\cdot)$ have linear growth in the space variable:

$$||f(x,u)||^2 + ||F(x,u)||^2 \leq K^2(1 + ||x||^2), \quad \forall\ x, y \in \mathbb{R}^{d_x},\ \ u \in U,$$

and the initial distribution of $x(0)$ is such that $\mathbb{E}[||x(0)||^2] \leq \infty$, then Eq. (2.7) *has a strong solution $x(\cdot)$ for this initial (random) initial condition $x(0)$. In both cases, a strong solution is also unique in the strong sense due to Theorem 2.2.8.*

The weak existence and uniqueness concepts allow for a more general class of drift vectors and dispersion matrices in controlled diffusion models.

Theorem 2.2.10 (Conditions for weak uniqueness and existence, see Theorems 5.3.10 and 5.4.22 in [128]) *When $f(\cdot,\cdot)$ and $F(\cdot,\cdot)$ are uniformly bounded, measurable, continuous functions, and the initial distribution of $x(0)$ is such that $\mathbb{E}[||x(0)||^2] \leq \infty$, then Eq.* (2.7) *has a weak solution that is unique in the weak sense.*

The boundedness requirement is naturally satisfied in many applications and is also needed for the implementation of the proposed Markov approximation method. These conditions can be *relaxed significantly* to allow for drift with *discontinuity* in the work of Kushner and Dupuis [43] when $x(\cdot)$ takes values in a bounded set. We will provide the details of these conditions in Chapter 3.

Remarkably, Kushner and Dupuis have shown that weak solutions that are unique in the weak sense and certain local properties are sufficient for the convergence of approximating solutions when solving stochastic optimal control problems [43]. We will present this important result in Section 3.2.

2.2.3 Geometric dynamic programming

We consider the controlled diffusion process in $S \subset \mathbb{R}^{d_x}$ in the previous subsection:

$$dx(t) = f(x(t), u(t))dt + F(x(t), u(t))dw(t).$$

In *a stochastic target problem*, we want to steer the process $x(\cdot)$ to a given stochastic target set $G \subset \mathbb{R}^{d_x}$ at time T by appropriately choosing a control process $u(\cdot)$. The reachability set $V(t)$ at time t is a set of all values of $x(t)$ such that $x(T) \in G$ almost surely for some admissible control process $u(\cdot)$:

$$V(t) = \{z \in \mathbb{R}^{d_x} \mid x(t) = z\ \wedge\ x(T) \in G \text{ a.s. for some admissible } u(\cdot)\}. \tag{2.9}$$

Historically, the evolution of reachability sets can be characterized by the geometric flows of their boundaries (see [131] and references therein). The following theorem, called geometric dynamic programming proposed and proven by Soner and Touzi, provides a stochastic representation for the evolution.

Theorem 2.2.11 (Geometric dynamic programming, see Theorem 3.1 in [131]) *Let $\tau \geq t$ be a stopping time. Then, we can relate $V(t)$ with $V(\tau)$ as follows:*

$$V(t) = \{z \in \mathbb{R}^{d_x} \mid x(t) = z \ \wedge \ x(\tau) \in V(\tau) \text{ a.s. for some admissible } u(\cdot)\}. \quad (2.10)$$

The relation in Eq. (2.10) resembles Bellman's dynamic programming principle for optimality, and hence the name. Intuitively, the principle asserts that a trajectory starting from a state in a time-t reachability set $V(t)$ will almost surely pass through any later time-τ reachability set $V(\tau)$ to reach the target set G.

We further assume that reachability sets have the following monotonicity property:

Assumption 2.2.12 (Monotonicity property, see [24]) *Let us consider a special case when $x(t)$ has two components $x(t) = (y, q)$ where $y \in \mathbb{R}^{d_x-1}$ and $q \in \mathbb{R}$. We say a reachability set $V(t)$ is monotonically increasing in q if $x(t) = (y, q) \in V(t)$ implies $x'(t) = (y, q') \in V(t)$ for all $q' > q$.*

Then, we define $\gamma : [0, \infty) \times \mathbb{R}^{d_x-1} \to \mathbb{R}$ as the infimum of the q component such that $x(t)$ belongs to the reachability set $V(t)$:

$$\gamma(t, y) = \inf\{\ q \in \mathbb{R} \mid (y, q) \in V(t)\ \}. \quad (2.11)$$

Under the monotonicity property, the geometric dynamic programming principle leads to the following results.

Theorem 2.2.13 (see [24]) *When reachability sets $V(t)$ are monotonically increasing in q, let $\tau > t$ be a stopping time, we have:*

- *If $q > \gamma(t, y)$, then there exists an admissible control $u(\cdot)$ that drives the process $x(\cdot)$ from $x(t) = (y, q)$ such that*

  $$q(\tau) \geq \gamma(\tau, y(\tau))$$

  happens almost surely.

- *If $q < \gamma(t, y)$, then for all admissible control $u(\cdot)$, starting from $x(t) = (y, q)$, we have:*

  $$P\big(q(\tau) > \gamma(\tau, y(\tau))\big) < 1.$$

In other words, there is no control process $u(\cdot)$ that will drive the process $x(t)$ to reach the reachability set $V(\tau)$, in full probability, when $x(\cdot)$ starts from a state $x(t)$ outside of the reachability set $V(t)$ where $t < \tau$.

2.2.4 Markov chains

A Markov chain is a discrete stochastic process $\{X_i;\ i \in \mathbb{N}\}$ with the property that given the present, future values are independent of the past:

$$P(X_{i+1} = x_{i+1} \mid X_i = x_i, X_{i-1} = x_{i-1}, ..., X_0 = x_0) = P(X_{i+1} = x_{i+1} \mid X_i = x_i).$$

We denote $P(X_{i+1} = x_{i+1} \mid X_i = x_i)$ shortly as $p(x_i, x_{i+1})$. A Markov chain takes value in a state space S, i.e. $X_i \in S$. A state $x \in S$ is called an absorbing state if

$$p(x,y) = \begin{cases} 1, & \text{if } x = y. \\ 0, & \text{otherwise.} \end{cases}$$

Definition 2.2.14 (Absorbing Markov chain) *An absorbing Markov chain $\{X_i;\ i \in \mathbb{N}\}$ is a Markov chain that has at least one absorbing state and every non-absorbing state can reach an absorbing state in finitely many steps.*

Starting from $X_0 = x_0$, a process $\{X_i;\ i \in \mathbb{N}\}$ is called absorbed if there is an index i such that X_i hits an absorbing state.

Theorem 2.2.15 (Probability of Absorption, see [132]) *In an absorbing Markov chain $\{X_i;\ i \in \mathbb{N}\}$, the probability that the processes will be absorbed is 1. That is, for any two non-absorbing states x and y:*

$$\lim_{i \to \infty} P(X_i = y \mid X_0 = x) = 0.$$

Thus, regardless of initial states, an absorbing Markov chain will reach an absorbing state eventually almost surely.

2.3 K-Nearest Neighbor Graphs

Random geometric graphs are defined as a collection of points in a metric space where edges are connected pairwise when certain conditions satisfied [101–104, 133, 134]. A useful random graph model, called k-nearest neighbor (kNN) graphs, considers edges between k nearest neighbors as defined below.

Definition 2.3.1 (Random k-nearest neighbor graph) *Let $d, k, n \in \mathbb{N}$. A random k-nearest neighbor graph $G^{near}(n,k)$ in a bounded set $S \subset \mathbb{R}^d$ is a graph with n vertices $\{X_1, X_2, ..., X_n\}$ that are independent and uniformly distributed random variable in S such that (X_i, X_j), $i \neq j$, is an edge if X_j is among the k nearest neighbors of X_i or vice versa.*

We also have directed kNN graphs that are similarly defined:

Definition 2.3.2 (Random directed k-nearest neighbor graph) *Let $d, k, n \in \mathbb{N}$. A random k-nearest neighbor graph $\overrightarrow{G}^{near}(n,k)$ in a bounded set $S \subset \mathbb{R}^d$ is a graph with n vertices $\{X_1, X_2, ..., X_n\}$ that are independent and uniformly distributed random variable in S such that (X_i, X_j), $i \neq j$, is a directed edge from X_i to X_j if X_j is among the k nearest neighbors of X_i.*

Many works in the literature consider random kNN graphs with vertices generated from a homogeneous Poisson point process. In particular, a Poisson random variable $Poisson(\lambda)$ with intensity λ takes value in $\mathbb{N}_0$ such that $P(Poisson(\lambda) = k) = \dfrac{e^{-\lambda}\lambda^k}{k!}$.

The mean of $Poisson(\lambda)$ is λ. A homogeneous Poisson point process of intensity λ in $\mathbb{R}^d$ is a *random countable set* of points $\mathcal{P}_\lambda^d \subset \mathbb{R}^d$ such that, for any measurable set $S_1, S_2 \subset \mathbb{R}^d$ and $S_1 \cap S_2 = \emptyset$, the number of points of $\mathcal{P}_\lambda^d$ in each set are *independent* Poisson variables, i.e., $|\mathcal{P}_\lambda^d \cap S_1| = Poisson(\lambda\mu(S_1))$ and $|\mathcal{P}_\lambda^d \cap S_2| = Poisson(\lambda\mu(S_2))$ where μ is the Lebesgue measure on $\mathbb{R}^d$. The main advantage of the Poisson point process is independence among counting random variables of disjoint subsets, which makes the proofs of claims on random kNN graphs much easier and more elegant. In contrast, when the number of points is given *a priori*, such independence property does not hold. The following Lemma relates the homogeneous Poisson point process with a set of independently and uniformly sampled points in S.

Lemma 2.3.3 (Restricted homogeneous Poisson point process [135]) *We consider $\{X_i\}_{i\in\mathbb{N}}$ as a sequence of points which are sampled independently and uniformly from a set $S \subset \mathbb{R}^d$. Let $Poisson(n)$ with intensity n, then $\{X_1, X_2, ..., X_{Poisson(n)}\}$ is the restriction to S of a homogeneous Poisson point process with intensity $n/\mu(S)$.*

We thus denote by $G^{near}(Poisson(n), k)$ and $\overrightarrow{G}^{near}(Poisson(n), k)$ as random kNN graphs and random directed kNN graphs with vertices $\{X_1, X_2, ..., X_{Poisson(n)}\}$.

A connected graph is a graph in which there is a path connecting any two vertices. Connectivity is an important property of random kNN graphs. The following theorem asserts a condition for connectivity in random kNN graphs.

Theorem 2.3.4 (Connectivity of random kNN graphs, see [134] and [103]) *Let $G^{near}(Poisson(n), k)$ and $\overrightarrow{G}^{near}(Poisson(n), k)$ be a random kNN graph and a random directed kNN graph in $S \subset \mathbb{R}^2$ having vertices generated by a homogeneous Poisson point process with intensity $n/\mu(S)$. Then, there exists a constant $a_2^c > 0$ and a constant $\vec{a}_2^c > 0$ such that:*

$$i. \quad \lim_{n\to\infty} P(\{G^{near}(Poisson(n), \lfloor a\log(n)\rfloor) \text{ is connected }\}) = \begin{cases} 1, & \text{if } a \geq a_2^c. \\ 0, & \text{otherwise.} \end{cases}$$

$$ii. \quad \lim_{n\to\infty} P(\{\overrightarrow{G}^{near}(Poisson(n), \lfloor a\log(n)\rfloor) \text{ is connected }\}) = \begin{cases} 1, & \text{if } a \geq \vec{a}_2^c. \\ 0, & \text{otherwise.} \end{cases}$$

That is, the connectivity property of random undirected and directed kNN graphs exhibits a phase transition and holds almost surely in the limit when edges are formed among $\Theta(\log(n))$ nearest neighbors in a graph with n vectices. The current estimates for the constant threshold are $0.3043 \leq a_2^c \leq 0.5139$ and $0.7209 \leq \vec{a}_2^c \leq 0.9967$. The results in Theorem 2.3.4 are also known to hold when the set S is in high dimensional space (see,e.g., [136]).

We remark that $G^{near}(Poisson(n), k)$ and $\overrightarrow{G}^{near}(Poisson(n), k)$ are good approximate models of $G^{near}(n, k)$ and $\overrightarrow{G}^{near}(n, k)$ for large n. Thus, we say that in the limit of n approaching ∞, random undirected and directed kNN graphs $G^{near}(n, k)$ and $\overrightarrow{G}^{near}(n, k)$ are connected asymptotically almost surely if $k = \Theta(\log(n))$.

Chapter 3

Stochastic Optimal Control: Formulation and Algorithm

In this chapter, we present the standard stochastic optimal control problem without risk constraints. We describe how the incremental Markov Decision Process (iMDP) algorithm constructs approximate solutions that are asymptotically-optimal in a suitable probabilistic sense. We then present the convergence analysis of the algorithm. Subsequently, we show experimental results on the robot motion planning and control problem of reaching a goal region while avoiding collision with obstacles in an uncertain environment.[1]

3.1 Problem Formulation

In this section, we first present a generic stochastic optimal control problem formulation. Subsequently, we discuss how the formulation extends the standard motion planning problem.

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 $\{\Omega, \mathcal{F}, P\}$. We define $\{\mathcal{F}_t; t \geq 0\}$ as the augmented filtration generated by the Brownian motion $w(\cdot)$. Let a control process $\{u(t); t \geq 0\}$ be a U-valued, measurable stochastic process also defined on the same probability space such that the pair $(u(\cdot), w(\cdot))$ is admissible [137]. Let the set of all such control processes be $\mathcal{U}$. Let $\mathbb{R}^{d_x \times d_w}$ denote the set of all d_x by d_w real matrices. We consider

[1] Part of the presented materials in this chapter have appeared in our previous papers [137, 138].

systems with dynamics described by a controlled diffusion process:

$$dx(t) = f(x(t), u(t))\, dt + F(x(t), u(t))\, dw(t), \forall t \geq 0 \tag{3.1}$$

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 also assume that the matrix $F(\cdot, \cdot)$ has full rank so that the convergence properties of the proposed algorithm hold as we will see in Theorem 3.2.3.[2] By Theorem 2.2.10, Eq. (3.1) has a unique weak sense solution. The continuity requirement of f and F can be relaxed with mild assumptions [43, 137] such that we still have a unique weak solution of Eq. (3.1) [128]. We will present these relaxed conditions in Section 3.2.

Policy and cost-to-go function

Markov controls are admissible controls that depend only on the current state, i.e., $u(t)$ is a function only of $x(t)$, for all $t \geq 0$. It is well known that in control problems with full state information, the best Markov control performs as well as the best admissible control (see, e.g., [128, 129]). A Markov control policy defined on S is represented by the function $\mu : S \to U$. The set of all policies is denoted by Π. Define the *first exit time* $T_\mu : \Pi \to [0, +\infty]$ under policy μ as

$$T_\mu = \inf \big\{ t : x(t) \notin S^o \text{ and Eq. (3.1) and } u(t) = \mu(x(t)) \big\}.$$

Intuitively, T_μ is the first time that the trajectory of the dynamical system given by Eq. (3.1) with $u(t) = \mu(x(t))$ hits the boundary ∂S of S. By definition, $T_\mu = +\infty$ if $x(\cdot)$ never exits S^o. Clearly, T_μ is a random variable. Then, the expected cost-to-go function under policy μ is a mapping from S to $\mathbb{R}$ defined as

$$J_\mu(z) = \mathbb{E}\left[\int_0^{T_\mu} \alpha^t \, g\big(x(t), \mu(x(t))\big)\, dt + \alpha^{T_\mu} h(x(T_\mu)) \;\big|\; x(0) = z \right],$$

where $g : S \times U \to \mathbb{R}$ 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 $C > 0$ such that

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

We will address the discontinuity of g and h in Section 3.2.

The *optimal cost-to-go function* $J^* : S \to \mathbb{R}$ is defined in the following optimization

[2] The full rank requirement of F can be relaxed as discussed on page 279 of [43].

problem:

$$\mathcal{OPT}1: \quad J^*(z) = \inf_{\mu \in \Pi} J_\mu(z) \textit{ for all } z \in S. \tag{3.2}$$

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 asymptotically optimal if the sequence of solutions returned from the algorithm converges to an optimal solution in probability as the number of samples approaches infinity. The sequence of solutions returned from asymptotically-optimal algorithms are thus called asymptotically-optimal.

In this chapter, 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 3.3, constructs an approximating discrete data structure for the continuous problem using an incremental sampling-based algorithm. The algorithm approximates the optimal cost-to-go function and an optimal policy in an anytime fashion. This sequence of approximations is guaranteed to converge uniformly in probability to the optimal cost-to-go function and to find an ϵ-optimal policy for an arbitrarily small non-negative ϵ as the number of samples approaches infinity.

Relationship with the standard motion planning problem

The standard robot motion planning problem of finding a collision-free trajectory that reaches a goal region for a deterministic dynamical system can be defined as follows (see, e.g., [52]). Let $\mathcal{X} \subset \mathbb{R}^{d_x}$ be a compact set. Let the open sets $\mathcal{X}_{\text{obs}}$ and $\mathcal{X}_{\text{goal}}$ denote the obstacle region and the goal region, respectively. Define the obstacle-free space as $\mathcal{X}_{\text{free}} := \mathcal{X} \setminus \mathcal{X}_{\text{obs}}$. Let $x_{\text{init}} \in \mathcal{X}_{\text{free}}$. Consider the deterministic dynamical system $\dot{x} = f(x(t), u(t))\, dt$, where $f : \mathcal{X} \times U \to \mathbb{R}^{d_x}$. The *feasible motion planning problem* is to find a measurable control input $u : [0, T] \to U$ such that the resulting trajectory $x(t)$ is collision free , i.e., $x(t) \in \mathcal{X}_{\text{free}}$ and reaches the goal region, i.e., $x(T) \in \mathcal{X}_{\text{goal}}$. The *optimal motion planning* problem is to find a measurable control input u such that the resulting trajectory x solves the feasible motion planning problem with minium trajectory cost.

The optimization problem $\mathcal{OPT}1$ extends the classical motion planning problem with stochastic dynamics as described by Eq. (3.1). Given a goal set $\mathcal{X}_{\text{goal}}$ and an obstacle set $\mathcal{X}_{\text{obs}}$, we define a state space S to be

$$S = \mathcal{X} \setminus (\mathcal{X}_{\text{goal}} \cup \mathcal{X}_{\text{obs}}),$$

and thus $\partial\mathcal{X}_{\text{goal}} \cup \partial\mathcal{X}_{\text{obs}} \cup \partial\mathcal{X} = \partial S$. Due to the nature of Brownian motion, under most policies, there is some non-zero probability that collision with an obstacle set will occur. However, to penalize collision with obstacles in the control design process, the cost of terminating by hitting the obstacle set, i.e., $h(z)$ for $z \in \partial\mathcal{X}_{\text{obs}}$, can be made arbitrarily high. Clearly, the higher this number is, the more conservative the resulting policy will be. Similarly, the terminal cost function on the goal set, i.e., $h(z)$ for $z \in \partial\mathcal{X}_{\text{goal}}$, can be set to a small value to encourage terminating by hitting the goal region.

Nevertheless, setting such cost values does not provide an automatic way to select control policies that respect certain safety criteria. We thus enforce a constraint that bounds the probability of collision from an initial state in Chapter 5 to address this concern. This problem is known as *chance-constrained optimization* in robotics.

Hamilton-Jacobi-Bellman equation

The stochastic optimal control problem formulated in $\mathcal{OPT}1$ have been studied widely in the literature. When the optimal cost-to-go function, or value function, J^* is differentiable at least twice, it is well-known that J^* is a solution of the following Hamilton-Jacobi-Bellman (HJB) equation:

$$\ln(\alpha)J(x) = \inf_{u \in U} \left[g(x,u) + \frac{\partial J^T(x)}{\partial x} f(x,u) + \frac{1}{2} Tr\left(F(x,u)F^T(x,u) \frac{\partial^2 J(x)}{(\partial x)^2} \right) \right], \ \forall x \in S^o, \quad (3.3)$$

with the boundary condition $J(x) = h(x)$ for $x \in \partial S$. Under the said smoothness condition, the above HJB equation can be derived from the Bellman's dynamic programming principle and Itô lemma.

Deriving similar equations for a broader class of problems, e.g., those with terminal state constraints and impulse control, is not always possible, and the optimal cost-to-go functions are usually not smooth enough to be classical solutions of the HJB equation. The Markov chain approximation method is a probabilistic approach that does not rely on deriving and solving HJB equations. In the next section, we present the main results from the Markov chain approximation method that will be used to prove the convergence of anytime solutions in our proposed algorithm.

3.2 The Markov Chain Approximation Method

The main idea of the Markov chain approximation method is to approximate the underlying system dynamics with a sequence of Markov chains such that it maintains certain local properties that are similar to those of the original system dynamics. Each Markov chain is defined on a Markov Decision Process (MDP) having an approximate cost function that is also analogous to the original cost function. Under very mild conditions, the sequence of optimal cost functions of approximating problems converges to the original optimal cost function as the approximation parameter goes to zero. In the following, we discuss this idea in detail.

Approximating Markov Decision Processes

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,
- $P(\cdot \mid \cdot, \cdot) : X \times X \times A \to \mathbb{R}_{\geq 0}$ is a function that denotes the transition probabilities satisfying $\sum_{\xi' \in X} P(\xi' \mid \xi, v) = 1$ for all $\xi \in X$ and all $v \in A$,
- $G(\cdot, \cdot) : X \times A \to \mathbb{R}$ is an immediate cost function, and
- $H : X \to \mathbb{R}$ is a terminal cost function.

If we start at time 0 with a state $\xi_0 \in X$, and at time $i \geq 0$, we apply an action $v_i \in A$ at a state ξ_i to arrive at a next state ξ_{i+1} according to the transition probability function P, we have a *controlled Markov chain* $\{\xi_i; i \in \mathbb{N}\}$. The chain $\{\xi_i; i \in \mathbb{N}\}$ due to the control sequence $\{v_i; i \in \mathbb{N}\}$ and an initial state ξ_0 will also be called the *trajectory* of $\mathcal{M}$ under the said sequence of controls and initial state.

Given a continuous-time dynamical system as described in Eq. (3.1), the Markov chain approximation method approximates the continuous stochastic dynamics using a sequence of MDPs $\{\mathcal{M}_n\}_{n=0}^{\infty}$ in which $\mathcal{M}_n = (S_n, U, P_n, G_n, H_n)$ where S_n is a discrete subset of S, and U is the original control set. We define the boundary ∂S_n of the finite state S_n as:

$$\partial S_n = \partial S \cap S_n.$$

For each $n \in \mathbb{N}$, let $\{\xi_i^n; i \in \mathbb{N}\}$ be a controlled Markov chain on $\mathcal{M}_n$ until it hits ∂S_n. We associate with each state z in S a non-negative interpolation interval $\Delta t_n(z)$, known as *a holding time*. We define

$$t_i^n = \sum_{0}^{i-1} \Delta t_n(\xi_i^n) \text{ for } i \geq 1 \text{ and } t_0^n = 0,$$

and

$$\Delta \xi_i^n = \xi_{i+1}^n - \xi_i^n.$$

Let u_i^n denote the control used at step i for the controlled Markov chain. In addition, we define the approximating cost rate and terminal cost functions as:

$$G_n(z, v) = g(z, v)\Delta t_n(z) \text{ and } H_n(z) = h(z) \text{ for each } z \in S_n \text{ and } v \in U. \tag{3.4}$$

A control problem for the MDP $\mathcal{M}_n$ is analogous to that defined in Section 3.1. Similar to previous section, 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. Given a policy μ_n, the (discounted) cost-to-go due to μ_n is:

$$J_{n,\mu_n}(z) = \mathbb{E}_{P_n}\left[\sum_{i=0}^{I_n - 1} \alpha^{t_i^n} G_n(\xi_i^n, \mu_n(\xi_i^n)) + \alpha^{t_{I_n}^n} H_n(\xi_{I_n}^n) \;\middle|\; \xi_0^n = z\right], \tag{3.5}$$

where $\mathbb{E}_{P_n}$ denotes the conditional expectation under P_n, the sequence $\{\xi_i^n; i \in \mathbb{N}\}$ is the controlled Markov chain under the policy μ_n, and I_n is the termination time defined as $I_n = \min\{i : \xi_i^n \in \partial S_n\}$.

The *optimal cost function*, denoted by $J_n^* : S_n \to \mathbb{R}$, is computed in the following optimization problem:

$$\mathcal{M}\_\mathcal{OPT}1 : \quad J_n^*(z) = \inf_{\mu_n \in \Pi_n} J_{n,\mu_n}(z), \quad \forall z \in S_n. \tag{3.6}$$

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 $||J_{n,\mu_n} - J_n^*||_\infty \leq \epsilon$. We call $\{u_i^n; i \in \mathbb{N}\}$ *a sequence of minimizing controls* if each control is an output of an ϵ-optimal policy for some $\epsilon > 0$.

We have presented a sequence of MDP problems $\mathcal{M}\_\mathcal{OPT}1$ that approximates the dynamics and objective function of the optimization problem $\mathcal{OPT}1$. Let us remark that the controlled Markov chains differ from the stochastic dynamical system described in Section 3.1 in that the former possesses a discrete state structure and evolves in a discrete time manner while the latter is a continuous model both in terms of its state space and the evolution of time. Yet, both models possess a continuous control space. We now relate the optimal cost-to-go J_n^* in approximating problems $\mathcal{M}\_\mathcal{OPT}1$ to the optimal cost-to-go J^* of $\mathcal{OPT}1$.

Previous convergence results

Intuitively, to have an approximating MDP sequence $\{\mathcal{M}_n\}_{n=0}^\infty$ that is consistent with the original continuous-time system dynamics, the MDPs should have similar local properties to the system dynamics. It turns out that only the mean and covariance of displacement per step along a Markov chain under any control are required to be close enough to those of the original dynamics so that desired convergence properties hold. These conditions are called *local consistency conditions* as below.

Definition 3.2.1 (Local consistency conditions) *Let Ω_n be the sample space of $\mathcal{M}_n$. Holding times Δt_n and transition probabilities P_n are said to be* locally consistent *with the dynamics in Eq.* (3.1) *if they satisfy the following conditions:*

1. *For all $z \in S$,*

$$\lim_{n\to\infty} \Delta t_n(z) = 0, \tag{3.7}$$

2. *For all $z \in S$ and all $v \in U$:*

$$\begin{aligned} \lim_{n\to\infty} \frac{\mathbb{E}_{P_n}[\Delta \xi_i^n \mid \xi_i^n = z, u_i^n = v]}{\Delta t_n(z)} &= f(z,v), & (3.8)\\ \lim_{n\to\infty} \frac{\mathrm{Cov}_{P_n}[\Delta \xi_i^n \mid \xi_i^n = z, u_i^n = v]}{\Delta t_n(z)} &= F(z,v)F(z,v)^T, & (3.9)\\ \lim_{n\to\infty} \sup_{i\in\mathbb{N},\omega\in\Omega_n} ||\Delta \xi_i^n||_2 &= 0. & (3.10) \end{aligned}$$

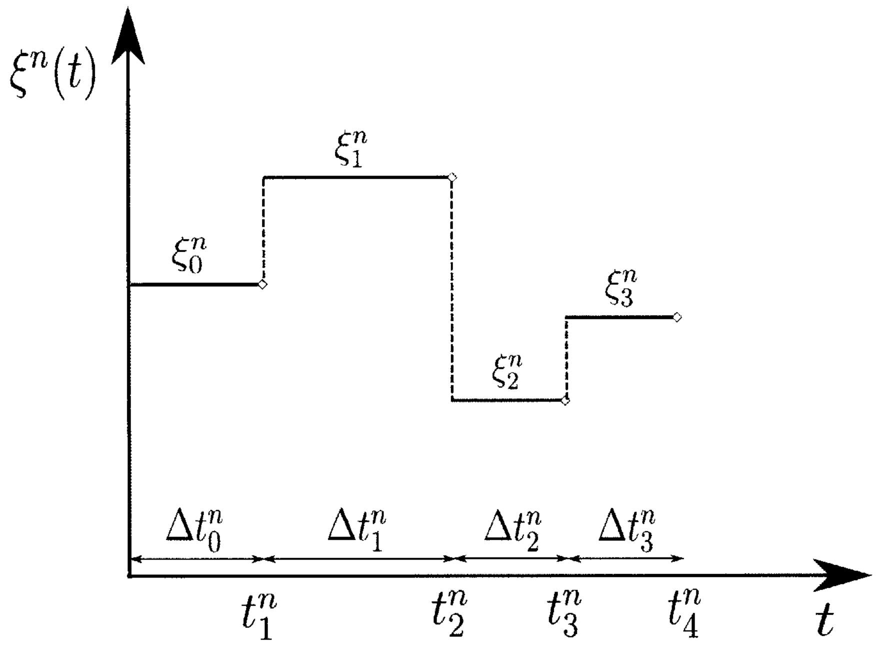


Figure 3-1: An illustration of a continuous-time interpolation of a discrete process $\{\xi_i^n; i \in \mathbb{N}\}$.

The chain $\{\xi_i^n; i \in \mathbb{N}\}$ is a discrete-time process. To show formal convergence to the continuous-time process $x(\cdot)$ in Eq. (3.1), we use an *approximate continuous-time interpolation* of the chain $\{\xi_i^n; i \in \mathbb{N}\}$. In particular, we define the (stochastic) continuous-time interpolation $\xi^n(\cdot)$ of the chain $\{\xi_i^n; i \in \mathbb{N}\}$ under the holding times function Δt_n as follows:

$$\xi^n(\tau) = \xi_i^n \textit{ for all } \tau \in [t_i^n, t_{i+1}^n).$$

Let $D^{d_x}[0, +\infty)$ denote the set of all $\mathbb{R}^{d_x}$-valued functions that are continuous from the left and has limits from the right. The process $\xi^n(\cdot)$ can be thought of as a random mapping from Ω_n to the function space $D^{d_x}[0, +\infty)$, and each realization of $\xi^n(\cdot)$ is a piece-wise constant function. This interpolation is described in Fig. 3-1. The continuous-time interpolation $u^n(\cdot)$ of the control sequence $\{u_i^n; i \in \mathbb{N}\}$ under the holding times function Δt_n is defined in a similar way:

$$u^n(\tau) = u_i^n \textit{ for all } \tau \in [t_i^n, t_{i+1}^n).$$

As stated in the following theorem, under mild technical assumptions, local consistency and the existence of a weakly unique solution of Eq. (3.1) together imply the *convergence in distribution* of the continuous-time interpolations of the trajectories of the controlled Markov chains to the trajectories of the stochastic dynamical system described by Eq. (3.1).

Theorem 3.2.2 (see Theorem 10.4.1 in [43]) *Let us assume that $f(\cdot, \cdot)$ and $F(\cdot, \cdot)$*

are measurable, bounded and continuous. Thus, Eq. (3.1) *has a weakly unique solution. Let* $\{\mathcal{M}_n\}_{n=0}^{\infty}$ *be a sequence of MDPs, and* $\{\Delta t_n\}_{n=0}^{\infty}$ *be a sequence of holding times that are locally consistent with the stochastic dynamical system described by Eq.* (3.1).

Let $\{u_i^n; i \in \mathbb{N}\}$ *be a sequence of controls defined for each* $n \in \mathbb{N}$. *For all* $n \in \mathbb{N}$, *let* $\{\xi^n(t); t \in \mathbb{R}_{\geq 0}\}$ *denote the continuous-time interpolation to the chain* $\{\xi_i^n; i \in \mathbb{N}\}$ *under the control sequence* $\{u_i^n; i \in \mathbb{N}\}$ *starting from an initial state* z_{init}, *and* $\{u^n(t); t \in \mathbb{R}_{\geq 0}\}$ *denote the continuous-time interpolation of* $\{u_i^n; i \in \mathbb{N}\}$, *according to the holding time* Δt_n.

Then, any subsequence of $\{(\xi^n(\cdot), u^n(\cdot))\}_{n=0}^{\infty}$ *has a further subsequence that converges in distribution to some limiting processes* $(x(\cdot), u(\cdot))$ *satisfying*

$$x(t) = z_{\text{init}} + \int_0^t f(x(\tau), u(\tau))d\tau + \int_0^t F(x(\tau), u(\tau))dw(\tau).$$

Under the weak uniqueness condition for solutions of Eq. (3.1), *the approximating sequence* $\{(\xi^n(\cdot), u^n(\cdot))\}_{n=0}^{\infty}$ *also converges in distribution to the limiting processes* $(x(\cdot), u(\cdot))$.

Effectively, Theorem 3.2.2 asserts a powerful result on the quality of approximation using the discrete-time discrete-state MDP data structure for the continuous-time continuous-space problem. Since the convergence is in distribution, simpler computation on discrete-state MDPs would allow us to approximate arbitrarily well the values of several variables in the continuous-time model. Indeed, *a sequence of minimizing controls* of approximating problems $\mathcal{M}\_\mathcal{OPT}1$ guarantees pointwise convergence of the cost function to the original optimal cost function of $\mathcal{OPT}1$ in the following sense.

Theorem 3.2.3 (see Theorem 10.5.2 in [43]) *Assume that* $f(\cdot,\cdot)$, $F(\cdot,\cdot)$, $g(\cdot,\cdot)$ *and* $h(\cdot)$ *are measurable, bounded and continuous. Let* $\{\mathcal{M}_n = (S_n, U, P_n, G_n, H_n)\}_{n=0}^{\infty}$ *and* $\{\Delta t_n\}_{n=0}^{\infty}$ *be locally consistent with the system described by Eq.* (3.1). *For any trajectory* $x(\cdot)$ *of the system described by Eq.* (3.1), *we define the first exit time on* $\mathcal{M}_n$ *as*

$$\hat{\tau}(x) := \inf\{t : x(t) \notin S^o\}.$$

We suppose that the function $\hat{\tau}(\cdot)$ *is continuous (as a mapping from* $D^{d_x}[0, +\infty)$ *to the compactified interval* $[0, +\infty]$*) with probability one relative to the measure induced by any solution of Eq.* (3.1) *for an initial state* z. *This assumption is satisfied when the matrix* $F(\cdot,\cdot)F(\cdot,\cdot)^T$ *is nondegenerate.*[3]

Then, for any $z \in S_n$, *the following equation holds:*

$$\lim_{n\to\infty} |J_n^*(z) - J^*(z)| = 0. \tag{3.11}$$

In particular, for any $z \in S_n$, *for any sequence* $\{\epsilon_n > 0\}_{n=0}^{\infty}$ *such that* $\lim_{n\to\infty} \epsilon_n = 0$, *and for any sequence of policies* $\{\mu_n\}_{n=0}^{\infty}$ *such that* μ_n *is an* ϵ_n*-optimal policy of* $\mathcal{M}_n$,

[3] Other conditions on f and F that satisfy this assumption are discussed on page 279 of [43].

we have:

$$\lim_{n\to\infty} |J_{n,\mu_n}(z) - J^*(z)| = 0. \tag{3.12}$$

Moreover, the sequence $\{t^n_{I_n}; n \in \mathbb{N}\}$ *converges in distribution to the termination time of the optimal control problem for the system in Eq.* (3.1) *when the system is under optimal control processes.*

Under the assumption that the cost rate g is Hölder continuous [139] with exponent 2ρ, the sequence of optimal value functions J^*_n for approximating chains indeed converges uniformly to J^* with a proven rate. Let us denote $||b||_{S_n} = \sup_{z\in S_n} b(x)$ as the sup-norm over S_n of a function b with domain containing S_n. Let

$$\zeta_n = \max_{z\in S_n} \min_{z'\in S_n} ||z' - z||_2 \tag{3.13}$$

be the dispersion of S_n. The following theorem asserts the uniform convergence of the sequence $\{J^*_n\}^\infty_{n=0}$ to J^*.

Theorem 3.2.4 (see Theorem 2.3 in [140] and Theorem 2.1 in [141]) *Consider an MDP sequence* $\{\mathcal{M}_n = (S_n, U, P_n, G_n, H_n)\}^\infty_{n=0}$ *and holding times* $\{\Delta t_n\}^\infty_{n=0}$ *that are locally consistent with the system described by Eq.* (3.1)*. Let* J^*_n *be the optimal cost of* $\mathcal{M}_n$*. Given the assumptions on the dynamics and cost rate functions in Section 3.1, as* n *approaches* ∞*, we have*

$$||J^*_n - J^*||_{S_n} = O(\zeta^\rho_n). \tag{3.14}$$

The details of the proofs for Theorems 3.2.2-3.2.3 can be found in the book of Kushner and Dupuis [43]. We remark that the proofs are purely probabilistic without appealing to regularity conditions for the optimal cost-to-go function. Similarly, the proof of Theorem 3.2.4 also relies on a probabilistic representation of value functions in terms of controlled Markov chains [141]. These proofs also provide insights into how to relax conditions on dynamics and cost functions. In particular, the above results still hold for functions f, F, g, h with discontinuity under mild technical conditions as below.

Discontinuity of dynamics and objective functions

When the functions $f, F, g,$ and h are discontinuous, the following conditions are sufficient to use Theorems 3.2.2-3.2.4:

(i) For r to be $f, F, g,$ or h, $r(x, u)$ takes either the form $r_0(x)+r_1(u)$ or $r_0(x)r_1(u)$ where the control dependent terms are continuous and the x-dependent terms are measurable, and

(ii) $f(x,\cdot), F(x,\cdot), g(x,\cdot)$, and $h(x)$ are nondegenerate for each x, and the set of discontinuity in x of each function is a uniformly smooth surface of lower dimension. Furthermore, instead of uniform Hölder continuity, the cost rate g can be relaxed to be locally Hölder continuous with exponent 2ρ on S (see, e.g., page 275 in [43] and page 720 in [141]).

Theorems 3.2.2-3.2.4 assert the asymptotic optimality given a sequence of *a priori* discretizations of the state space and the availability of ϵ-optimal policies. Thus, we need to solve the optimization problem $\mathcal{M}\_\mathcal{OPT}1$ for each $n \in \mathbb{N}$ to obtain an ϵ-optimal policy for the MDP $\mathcal{M}_n$. This can be done using the value iteration or policy iteration algorithms on successive grid discretization of the state space S. However, solving $\mathcal{M}\_\mathcal{OPT}1$ repeatedly is computationally challenging due to the curse of dimensionality, especially when the number of states grows rapidly over iterations.

In what follows, we describe an algorithm that incrementally computes the optimal cost-to-go function J^* and an optimal control policy μ^* of the continuous problem without directly computing the optimal cost-to-go function J_n^* and optimal policies μ_n^* for each approximating problem.

3.3 Incremental Markov Decision Process (iMDP) Algorithm

Based on the Markov chain approximation results, the iMDP algorithm incrementally builds a sequence of discrete MDPs with probability transitions and cost-to-go functions that consistently approximate the original continuous counterparts. Using the rapidly-exploring sampling technique [51] to sample in the state space, iMDP forms the structures of finite-state MDPs *randomly* over iterations. Control sets for states in these MDPs are constructed or sampled properly in the control space. The algorithm refines the discrete models by using a number of primitive procedures to add new states into the current approximating model. Finally, the algorithm improves the quality of discrete-model policies in an iterative manner by effectively using the computations inherited from the previous iterations.

3.3.1 Primitive procedures

Before presenting the algorithm, some primitive procedures which the algorithm relies on are presented in this subsection.

Sampling

The procedures `Sample`() and `SampleBoundary`() sample states independently and uniformly from the interior S^o and the boundary ∂S, respectively. We assume in this thesis that samples are drawn from a uniform distribution. However, different distributions, e.g. those with density bounded away from zero on S, can be used. When the geometric shapes of S and ∂S are complex, we can use rejection sampling with the help of a feasibility testing procedure.

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 a given distance function. Many choice of distance functions have been discussed in the work by LaValle and Kuffner [71]. In this work, we use the Euclidean norm as a distance function for simplicity.

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.[4] The parameter $\rho \in (0, 0.5]$ defines the Hölder continuity of the cost rate function $g(\cdot, \cdot)$ as in Section 3.1.

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

- 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
- 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 [43]. In particular, we choose

$$Z_{\text{near}} = \texttt{Nearest}(z + f(z, v)\tau, Y, s),$$

where $s = \Theta(\log(|Y|))$ so that Z_{near} has about $\log(|Y|)$ states. We define the transition probabilities $p : Z_{\text{near}} \to \mathbb{R}_{\geq 0}$ that satisfies:

(i) $\sum_{z' \in Z_{\text{near}}} p(z')(z' - z) = f(z, v)\tau + o(\tau)$,

(ii) $\sum_{z' \in Z_{\text{near}}} p(z')(z' - z)(z' - z)^T = F(z, v)F(z, v)^T \tau + f(z, v)f(z, v)^T \tau^2 + o(\tau)$.

(iii) $\sum_{z' \in Z_{\text{near}}} p(z') = 1$.

[4] Typical values of ς is [0.999,1). The role of this value will be clear in our convergence proofs.

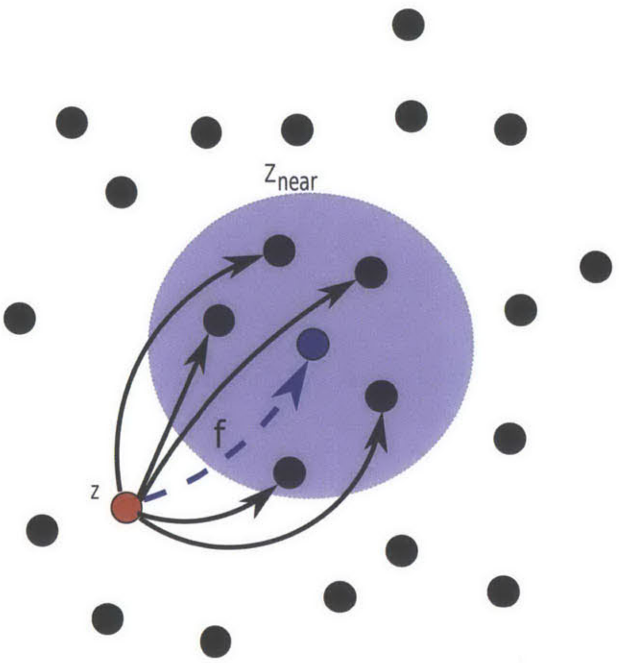


Figure 3-2: An illustration of transition probability construction. From a state z (red), we simulate the nominal dynamics (blue arrow) to get a new state $z + f(z, v)\tau$. The support Z_{near} that contains nodes around $z + f(z, v)\tau$ is shaded. Possible transitions from z to nodes in the support are represented by black arrows. Probabilities associated with these transitions are computed to satisfy the local consistency conditions.

An alternate way to compute the transition probabilities is to approximate using local Gaussian distributions. We also 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.

We note that solving a system of linear equations requires computing and handling a matrix of size $(d_x^2 + d_x + 1) \times |Z_{\text{near}}|$. In contrast, computing local Gaussian approximation requires only $|Z_{\text{near}}|$ evaluations. Thus, local Gaussian approximation provides lower time complexity and is the main method to construct locally consistent transition probabilities in this work.

Figure 3-2 shows an illustration of how the procedure `ComputeTranProb` constructs transition probabilities. As we can see, from a state z (red), we simulate the nominal dynamics (dash blue arrow) to get a new state $z + f(z, v)\tau$ (blue). The support Z_{near}

that contains nodes around $z + f(z, v)\tau$ is shaded, and possible transitions from z to the support nodes are represented by black arrows. Probabilities associated with these transitions are computed to satisfy the local consistency conditions as we discussed above.

Backward Extension

Given $T > 0$ and two states $z, z' \in S$, the procedure `ExtendBackwards`(z, z', T) returns a triple (x, v, τ) consisting of a trajectory, a control input, and a final time such that

- $dx(t) = f(x(t), u(t))dt$ and $u(t) = v \in U$ for all $t \in [0, \tau]$,
- Final time $\tau \leq T$, and $x(t) \in S$ for all $t \in [0, \tau]$,
- $x(\tau) = z$, and $x(0)$ is close to z',
- $\{x(t); t \in [0, \tau)\} \subset S^o$.

The last condition requires that except the terminal state z, the trajectory $x(\cdot)$ must remain in the interior of S. 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 using a feasibility test to choose the control resulting in a feasible trajectory $x(\cdot)$ with $x(0)$ that is closest to z'.[5]

Sampling and Discovering Controls

The procedure `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 $z' \in$ `Nearest`(z, Y, k), we solve for a control $v \in U$ such that

- $dx(t) = f(x(t), u(t))dt$ and $u(t) = v \in U$ for all $t \in [0, T]$,
- $x(t) \in S^o$ for all $t \in [0, T]$,
- $x(0) = z$ and $x(T) = z'$.

Using these primitive procedures, we now describe the iMDP algorithm in detail.

3.3.2 iMDP algorithm description

The iMDP algorithm is given in Algorithm 1. 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

[5] This procedure is used in the algorithm solely for the purpose of inheriting the "rapid exploration" property of the RRT algorithm [51,52]. The feasibility test is similar to the collision-checking procedure of the RRT algorithm.

time function Δt_n that consistently approximates the system in Eq. (3.1). In particular, given a state $z \in S_n$ and a holding time $\Delta t_n(z)$, we can implicitly define the stage cost function

$$G_n(z, v) = \Delta t_n(z) g(z, v)$$

for all $v \in U$, and the terminal cost function

$$H_n(z) = h(z).$$

We also associate with $z \in S_n$ a cost value $J_n(z)$, and a control $\mu_n(z)$. We refer to J_n as a cost value function over S_n. In the following discussion, we describe how to construct S_n, P_n, J_n, μ_n over iterations. We note that, in most cases, we only need to construct and access P_n on demand.

In every iteration of the main loop (Lines 4-12), we sample an additional state from the boundary of the state space S. We set $J_n, \mu_n, \Delta t_n$ for those states at Line 5. Subsequently, we also sample a state from the interior of S (Line 6) denoted as z_s. We compute the nearest state z_{nearest}, which is already in the current MDP, to the sampled state (Line 7). The algorithm computes a trajectory that reaches z_{nearest} starting at some state near z_s (Line 8) using a control signal $u_{\text{new}}(0..\tau)$. The new trajectory is denoted by $x_{\text{new}} : [0, \tau] \to S$ and the starting state of the trajectory, i.e., $x_{\text{new}}(0)$, is denoted by z_{new}. The new state z_{new} is added to the state set, and the cost value $J_n(z_{\text{new}})$, control $\mu_n(z_{\text{new}})$, and holding time $\Delta t_n(z_{\text{new}})$ are initialized at Line 11.

Update of cost value and control

The algorithm updates the cost values and controls of the finer MDP in Lines 13-15. We perform $L_n \geq 1$ value iterations in which we update the new state z_{new} and other $K_n = \Theta(|S_n|^\theta)$ states in the state set where $K_n < |S_n|$. When all states in the MDP are updated, i.e. $K_n + 1 = |S_n|$, L_n value iterations are implemented in a synchronous manner. Otherwise, L_n value iterations are implemented in an asynchronous manner.

The set of states to be updated is denoted as Z_{update} (Line 13). To update a state $z \in Z_{\text{update}}$ that is not on the boundary, in the call to the procedure `Update` (Line 15), we solve the following Bellman equation:[6]

$$J_n(z) = \min_{v \in U} \{G_n(z, v) + \alpha^{\Delta t_n(z)} \mathbb{E}_{P_n} \left[J_{n-1}(y) | z, v \right]\}, \tag{3.15}$$

and set $\mu_n(z) = v^*(z)$, where $v^*(z)$ is the minimizing control of the above optimization problem.

There are several ways to solve Eq. (3.15) over the the continuous control space U efficiently. If $P_n(\cdot \mid z, v)$ and $g(z, v)$ are affine functions of v, and U is convex, the above optimization has a linear objective function and a convex set of constraints.

[6] Although the argument of `Update` at Line 15 is J_n, we actually process the previous cost values J_{n-1} due to Line 3. We can implement Line 3 by simply sharing memory for $(S_n, J_n, \mu_n, \Delta t_n)$ and $(S_{n-1}, J_{n-1}, \mu_{n-1}, \Delta t_{n-1})$.

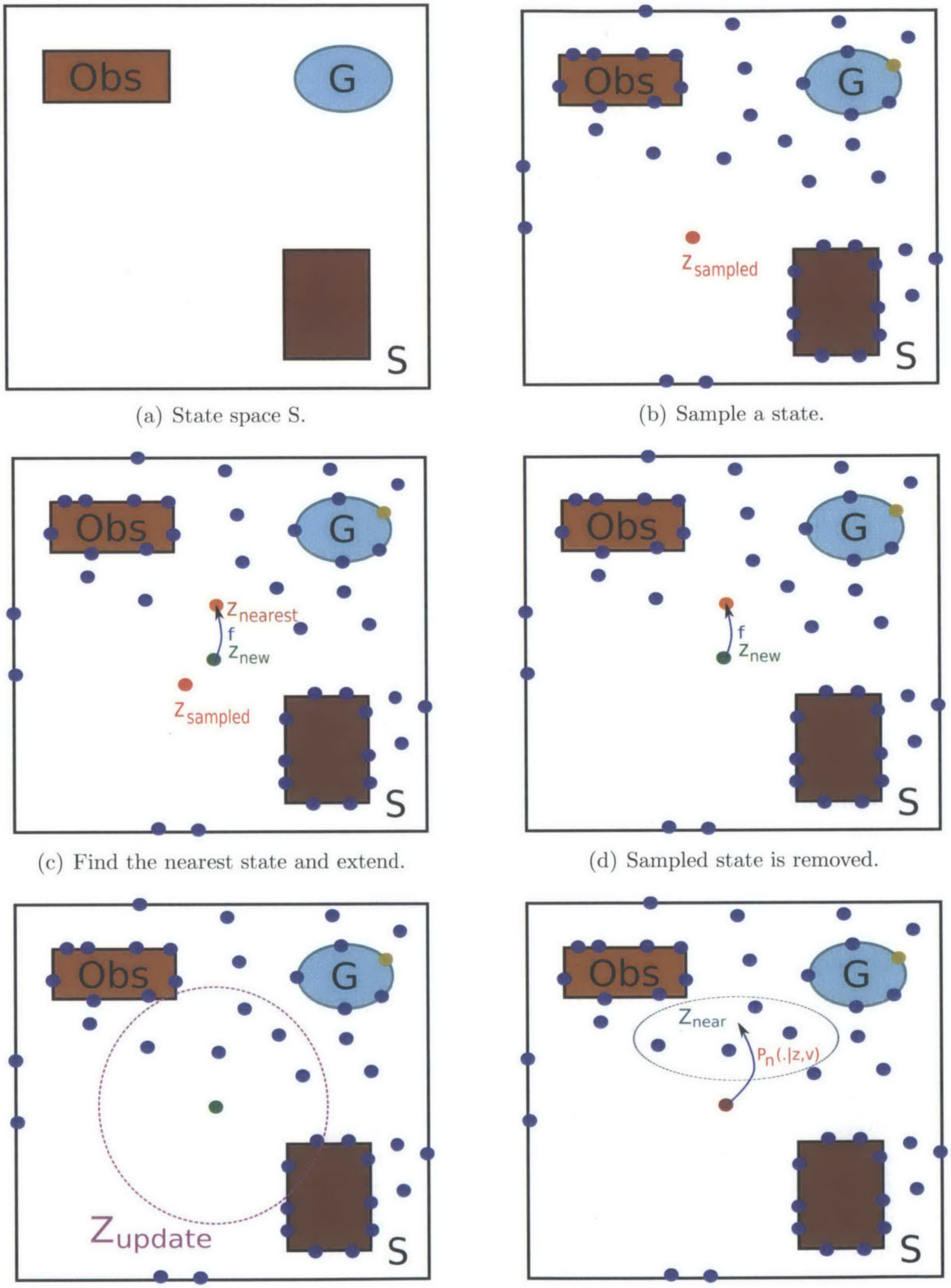
```
Algorithm 1: iMDP()
$(n, S_0, J_0, \mu_0, \Delta t_0) \leftarrow (1, \emptyset, \emptyset, \emptyset, \emptyset)$;
for $n = 1 \rightarrow N$ do
    $(S_n, J_n, \mu_n, \Delta t_n) \leftarrow (S_{n-1}, J_{n-1}, \mu_{n-1}, \Delta t_{n-1})$;
    // Add a new state to the boundary
    $z_{\mathrm{s}} \leftarrow$ SampleBoundary();
    $(S_n, J_n(z_{\mathrm{s}}), \mu_n(z_{\mathrm{s}}), \Delta t_n(z_{\mathrm{s}})) \leftarrow (S_n \cup \{z_{\mathrm{s}}\}, h(z_{\mathrm{s}}), null, 0)$ ;
    // Add a new state to the interior
    $z_{\mathrm{s}} \leftarrow$ Sample();
    $z_{\mathrm{nearest}} \leftarrow$ Nearest($z_{\mathrm{s}}, S_n, 1$);
    if $(x_{\mathrm{new}}, u_{\mathrm{new}}, \tau) \leftarrow$ ExtendBackwards($z_{\mathrm{nearest}}, z_{\mathrm{s}}, T_0$) then
        $z_{\mathrm{new}} \leftarrow x_{new}(0)$;
        $cost = \tau g(z_{\mathrm{new}}, u_{\mathrm{new}}) + \alpha^{\tau} J_n(z_{\mathrm{nearest}})$;
        $(S_n, J_n(z_{\mathrm{new}}), \mu_n(z_{\mathrm{new}}), \Delta t_n(z_{\mathrm{new}})) \leftarrow (S_n \cup \{z_{\mathrm{new}}\}, cost, u_{new}, \tau)$ ;
        // Perform $L_n \geq 1$ (asynchronous) value iterations
        for $i = 1 \rightarrow L_n$ do
            // $K_n = \Theta(|S_n|^{\theta})$ where $(0 < \theta \leq 1,\ K_n < |S_n|)$
            $Z_{\mathrm{update}} \leftarrow$ Nearest($z_{\mathrm{new}}, S_n \backslash \partial S_n, K_n$) $\cup \{z_{\mathrm{new}}\}$;
            for $z \in Z_{\mathrm{update}}$ do
                Update($z, S_n, J_n, \mu_n, \Delta t_n$);
```

Such problems are widely studied in the literature [142].

More generally, we can uniformly sample the set of controls, called U_n, in the control space U. Hence, we can evaluate the right hand side (RHS) of Eq. (3.15) for each $v \in U_n$ to find the best v^* in U_n with the smallest RHS value and thus to update $J_n(z)$ and $\mu_n(z)$. When $\lim_{n \to \infty} |U_n| = \infty$, we can solve Eq. (3.15) arbitrarily well (see Theorem 3.4.6).

Thus, it is sufficient to construct the set U_n with $\Theta(\log(|S_n|))$ controls using the procedure ConstructControls as described in Algorithm 2 (Line 2). The set Z_{near} and the transition probability $P_n(\cdot \,|\, z, v)$ constructed consistently over the set Z_{near} are returned from the procedure ComputeTranProb for each $v \in U_n$ (Line 4). Subsequently, the procedure chooses the best control among the constructed controls to update $J_n(z)$ and $\mu_n(z)$ (Line 7). We note that in Algorithm 2, before making improvement for the cost value at z by comparing new controls, we can re-evaluate the cost 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. The reason is that the current control may be still the best control compared to other controls in U_n.

The steps of the iMDP algorithms are illustrated in Fig. 3-3 using a motion planning problem in a two-dimensional state space as an example. We note that in this example, the state space S includes boundaries of obstacle regions and a goal region.



(a) State space S.

(b) Sample a state.

(c) Find the nearest state and extend.

(d) Sampled state is removed.

(e) Choose states to update.

(f) Compute locally consistent update.

Figure 3-3: Steps of the iMDP algorithm.

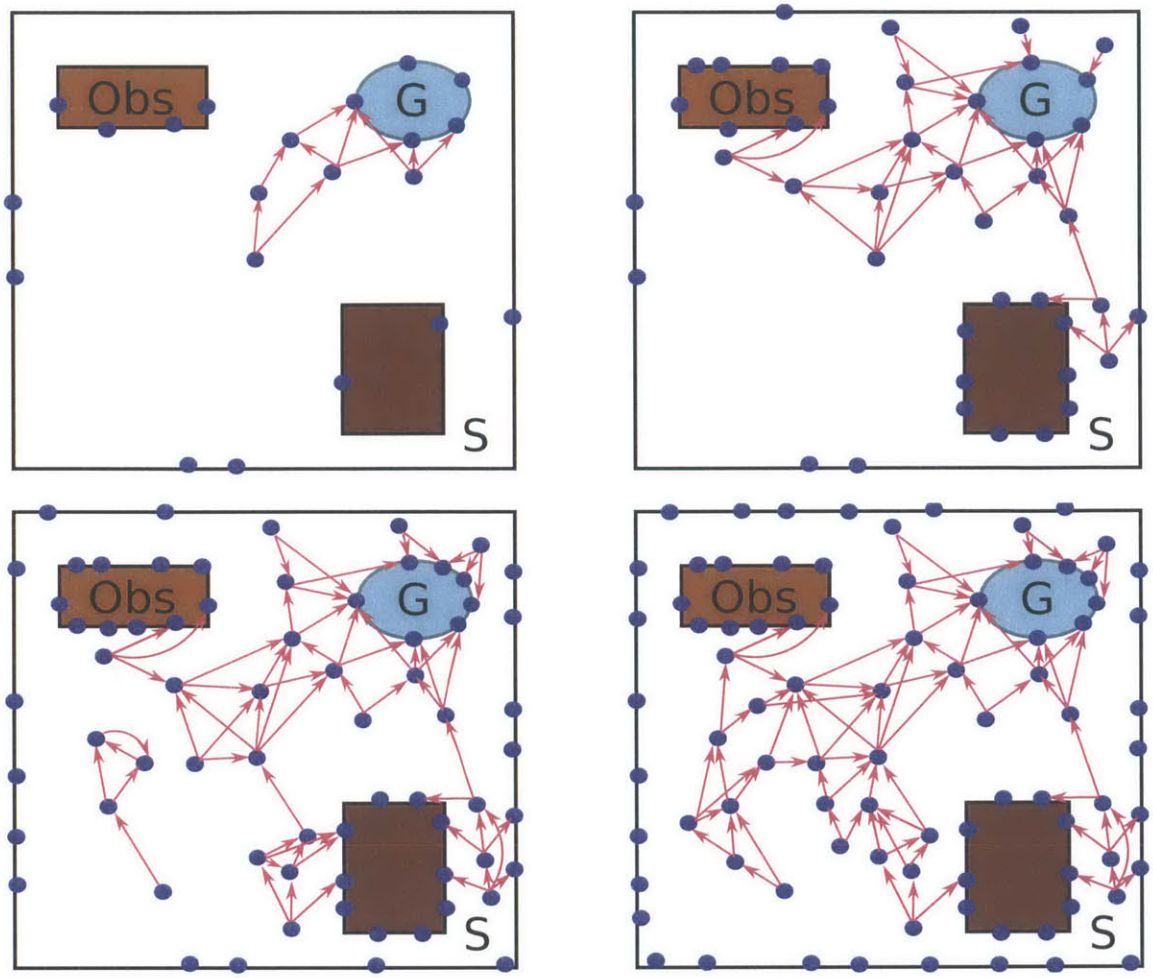


Figure 3-4: An illustration of Markov chains over iterations. States on boundary connect to themselves.

In Fig. 3-4, we show an example of how Markov chains, which are formed by following best control $\mu_n(z)$ to transit to states in S_n, look like over iterations. States on the boundary connect to themselves, and these links are not depicted. In the following analysis, we will characterize the connectivity of these Markov chains.

3.3.3 Complexity of iMDP

The time complexity per iteration of the Algorithms 1-2 is $O\big(|S_n|^\theta(\log|S_n|)^2\big)$ where θ is a parameter in $(0, 1]$. This is due to $\Theta(|S_n|^\theta)$ states that are updated in each iteration using $\Theta(\log(|S_n|))$ controls and transition probability functions with support size $\Theta(\log(|S_n|))$.

Since we only need to access locally consistent transition probability on demand, the space complexity of the iMDP algorithm is $O(|S_n|)$. Finally, the size of state space S_n is $|S_n| = \Theta(n)$ due to our sampling strategy.

```
Algorithm 2: Update($z \in S_n, S_n, J_n, \mu_n, \Delta t_n$)
$\tau \leftarrow$ ComputeHoldingTime($z, |S_n|$);
// Sample or discover $C_n = \Theta(\log(|S_n|))$ controls
$U_n \leftarrow$ ConstructControls($C_n, z, S_n, \tau$);
for $v \in U_n$ do
    $(Z_{\text{near}}, p_n) \leftarrow$ ComputeTranProb($z, v, \tau, S_n$);
    $J \leftarrow \tau g(z, v) + \alpha^{\tau} \sum_{y \in Z_{\text{near}}} p_n(y) J_n(y)$;
    if $J < J_n(z)$ then
        $(J_n(z), \mu_n(z), \Delta t_n(z)) \leftarrow (J, v, \tau, |S_n|)$;
```

```
Algorithm 3: Policy($z \in S, n$)
$z_{\text{nearest}} \leftarrow$ Nearest($z, S_n, 1$);
return $(\mu(z) = \mu_n(z_{\text{nearest}}), \Delta t_n(z_{\text{nearest}}))$
```

The comparison of iMDP with other sampling-based algorithms such as RRT and RRT* is shown in Table 3.1. As we can see, iMDP has the same space complexity as other algorithms. While iMDP spends a little more time per iteration, the algorithm can properly handle process noise and provide closed loop control policies.

3.3.4 Feedback control

As we will see in Theorems 3.4.5-3.4.6, the sequence of cost value functions J_n arbitrarily approximates the original optimal cost-to-go J^*. Therefore, 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 3.4.7 that the sequence of μ_n also approximates arbitrarily well an optimal control policy. In other words, in the iMDP algorithm, we also incrementally construct an optimal control policy. In the following paragraph, we present

Table 3.1: Comparison of sampling-based algorithms: RRT, RRT*, iMDP

	RRT	**RRT***	**iMDP**
Iteration Time Complexity	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$	$\mathcal{O}(n^{\theta}(\log n)^2)$
Space Complexity	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$
Asymptotic Optimality		✓	✓
Handling Process Noise			✓
Closed Loop Control			✓

an algorithm that converts a policy of a discrete system to a policy of the original continuous problem.

Given a level of approximation $n \in \mathbb{N}$, the control policy μ_n generated by the iMDP algorithm is used for controlling the original system described by Eq. (3.1) using the procedure given 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.

3.4 Analysis

In this section, we carry out the detailed convergence analysis of the iMDP algorithm. The proofs of the presented lemmas and theorems in this section can be found in Section 3.6. Throughout our analysis, let us denote $\big(\mathcal{M}_n = (S_n, U, P_n, G_n, H_n), \Delta t_n, J_n, \mu_n\big)$ as the MDP, holding times, cost value function, and policy returned by Algorithm 1 at the end n iterations.

First, we claim that Markov chains defined on $\mathcal{M}_n$ are absorbing Markov chains.

Theorem 3.4.1 *Let $\{\xi_i^n; i \in \mathbb{N}\}$ be a Markov chain on $\mathcal{M}_n$ formed by following the transition probabilities P_n using the best control $\mu_n(z)$ for each state $z \in S_n$. Then, $\{\xi_i^n; i \in \mathbb{N}\}$ is an absorbing Markov chain asymptotically almost surely.*

The proof follows from Theorem 2.3.4 on connectivity of random directed kNN graphs. Therefore, asymptotically almost surely, a controlled Markov chain on $\mathcal{M}_n$ will reach an absorbing state in the boundary set ∂S_n (see Theorem 2.2.15). In other words, the iMDP algorithm constructs approximating MDPs that induce random graphs for the effective exploration of the continuous search space S. We now show that this approximation is also consistent.

For large n, states in S_n are sampled uniformly in the state space S as proven in [52]. Moreover, the dispersion of S_n shrinks with the rate $O((\log|S_n|/|S_n|)^{1/d_x})$ as described in the next lemma.

Lemma 3.4.2 *Recall that ζ_n measures of the dispersion of S_n (Eq. (3.13)). We have the following event happens with probability one:*

$$\zeta_n = O((\log|S_n|/|S_n|)^{1/d_x}).$$

The proof is based on the fact that, if we partition $\mathbb{R}^{d_x}$ into cells of volume $O(\log(|S_n|)/|S_n|)$, then, almost surely, every cell contains at least an element of S_n, as $|S_n|$ approaches infinity. The above lemma leads to the following results.

Lemma 3.4.3 *The MDP sequence $\{\mathcal{M}_n\}_{n=0}^{\infty}$ and holding times $\{\Delta t_n\}_{n=0}^{\infty}$ returned by Algorithm 1 are locally consistent with the system described by Eq. (3.1) for large n with probability one.*

Theorem 3.2.2 and Lemma 3.4.3 together imply that the trajectories of the controlled Markov chains on $\{\mathcal{M}_n\}_{n=0}^{\infty}$ approximate, in the distribution sense, those of the original stochastic dynamical system in Eq. (3.1) arbitrarily well as n approaches to infinity.

Moreover, recall that $||\cdot||_{S_n}$ is the sup-norm over S_n, the following theorem shows that J_n^* converges uniformly, with probability one, to the original optimal value function J^*.

Theorem 3.4.4 *Given $n \in \mathbb{N}$, for all $z \in S_n$, $J_n^*(z)$ denotes the optimal value function evaluated at state z for the finite-state MDP $\mathcal{M}_n$ returned by Algorithm 1. Then, the following event holds with probability one:*

$$\lim_{n\to\infty} ||J_n^* - J^*||_{S_n} = 0.$$

In other words, J_n^* converges to J^* uniformly almost surely. In particular,

$$||J_n^* - J^*||_{S_n} = O((\log |S_n|/|S_n|)^{\rho/d_x}) \ a.s.$$

The proof of Theorem 3.4.4 follows immediately from Lemmas 3.4.2-3.4.3 and Theorems 3.2.3-3.2.4. The theorem suggests that we can compute J_n^* for each discrete MDP $\mathcal{M}_n$ before sampling more states to construct $\mathcal{M}_{n+1}$. Indeed, in Algorithm 1, when updated states are chosen randomly as subsets of S_n, and L_n is large enough, we compute J_n^* using asynchronous value iterations [33, 41]. Subsequent theorems present *stronger* results on incremental computation of J^*.

We will prove the asymptotic optimality of the cost value J_n returned by the iMDP algorithm when n approaches infinity without directly approximating J_n^* for each n. We first consider the case when we can solve the Bellman update (Eq. (3.15)) exactly and $1 \leq L_n,\ K_n = \Theta(|S_n|^\theta) < |S_n|$.

Theorem 3.4.5 *For all $z \in S_n$, $J_n(z)$ is the cost value of the state z computed by Algorithm 1 and Algorithm 2 after n iterations with $1 \leq L_n$, and $K_n = \Theta(|S_n|^\theta) < |S_n|$. Let J_{n,μ_n} be the cost-to-go function of the returned policy μ_n on the discrete MDP $\mathcal{M}_n$. If the Bellman update at Eq. (3.15) is* ***solved exactly****, then, the following events hold with probability one:*

i. $\lim_{n\to\infty} ||J_n - J_n^*||_{S_n} = 0$, *and* $\lim_{n\to\infty} ||J_n - J^*||_{S_n} = 0$,

ii. $\lim_{n\to\infty} |J_{n,\mu_n}(z) - J^*(z)| = 0, \quad \forall z \in S_n$.

Theorem 3.4.5 enables an incremental computation of the optimal cost J^* without the need to compute J_n^* exactly before sampling more samples. Moreover, cost-to-go functions J_{n,μ_n} induced by approximating policies μ_n also converges pointwise to the optimal cost-to-go J^* with probability one.

When we solve the Bellman update (Eq. (3.15)) via sampling, the next result holds.

Theorem 3.4.6 *For all $z \in S_n$, $J_n(z)$ is the cost value of the state z computed by Algorithm 1 and Algorithm 2 after n iterations with $1 \leq L_n$, and $K_n = \Theta(|S_n|^\theta) <$*

$|S_n|$. *Let* J_{n,μ_n} *be the cost-to-go function of the returned policy* μ_n *on the discrete MDP* $\mathcal{M}_n$. *If the Bellman update at Eq.* (3.15) *is* ***solved via sampling*** *such that* $\lim_{n\to\infty} |U_n| = \infty$, *then*

i. $||J_n - J_n^*||_{S_n}$ *converges to* 0 *in probability. Thus,* J_n *converges uniformly to* J^* *in probability,*

ii. $\lim_{n\to\infty} |J_{n,\mu_n}(z) - J^*(z)| = 0$ *for all* $z \in S_n$ *with probability one.*

We emphasize that while the convergence of J_n to J^* is weaker than the convergence in Theorem 3.4.5, the convergence of J_{n,μ_n} to J^* remains intact. Importantly, Theorem 3.2.2 and Theorems 3.4.5-3.4.6 together assert that starting from any initial state, trajectories and control processes provided by the iMDP algorithm approximate arbitrarily well optimal trajectories and optimal control processes of the original continuous problem. More precisely, with probability one, the induced random probability measures of approximating trajectories and approximating control processes converge weakly (in distribution) to the probability measures of optimal trajectories and optimal control processes of the continuous problem. In addition, Theorem 3.4.4 provides a proven convergence rate of J_n^* to J^*, which suggests that J_n is likely to converge to J^* with the same convergence rate $O((\log|S_n|/|S_n|)^{\rho/d_x})$.

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

Theorem 3.4.7 *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:*

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

Then there exists an optimal control policy μ^* *of the original problem*[7] *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.

3.5 Experiments

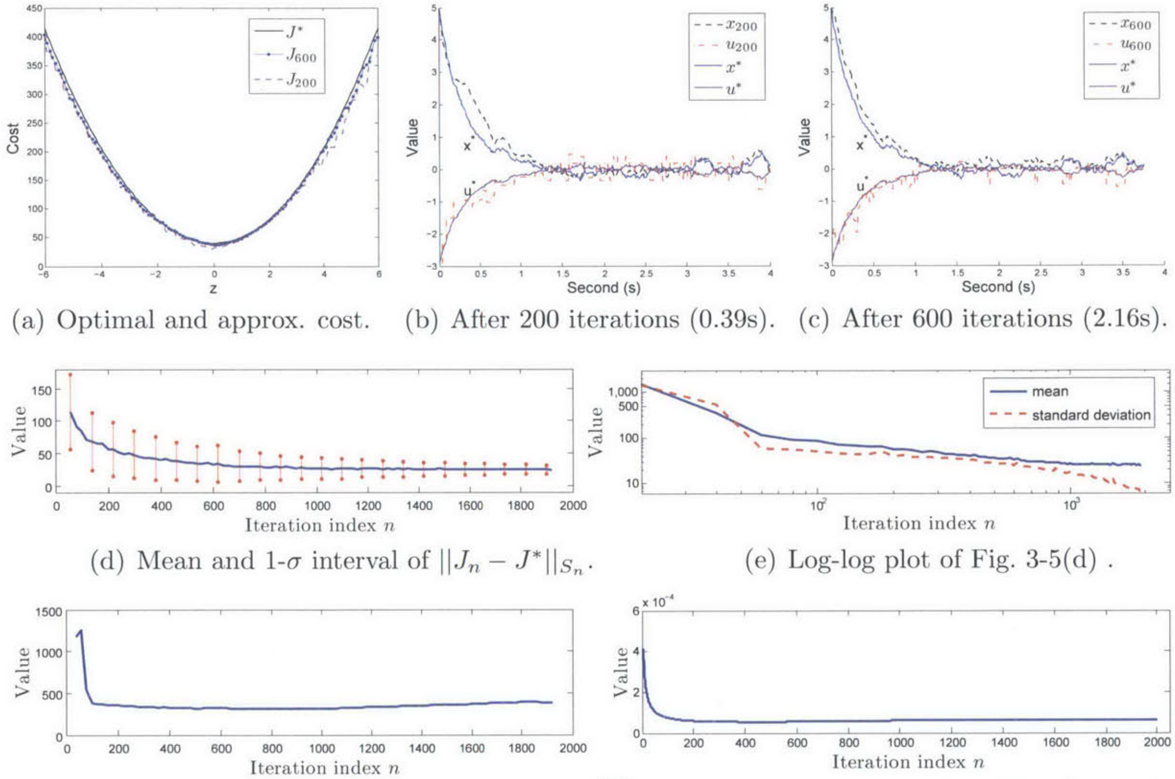
We used a computer with a 2.0-GHz Intel Core 2 Duo T6400 processor and 4 GB of RAM to run experiments. In the first experiment, we investigated the convergence of the iMDP algorithm on a stochastic LQR problem:

$$\inf_{u(\cdot)} \mathbb{E}\Big[\int_0^T 0.95^t\{3.5x(t)^2 + 200u(t)^2\}dt + 0.95^T h(x(T))\Big]$$

such that

$$dx(t) = (3x(t) + 11u(t))dt + \sqrt{0.2}dw(t)$$

[7] Otherwise, an optimal relaxed control policy m^* exists [43], and $\overline{\mu}_n$ approximates m^* arbitrarily well.



(a) Optimal and approx. cost. (b) After 200 iterations (0.39s). (c) After 600 iterations (2.16s).

(d) Mean and 1-σ interval of $||J_n - J^*||_{S_n}$. (e) Log-log plot of Fig. 3-5(d) .

(f) Plot of ratio $||J_n - J^*||_{S_n} / \big(\log(|S_n|)/|S_n|\big)^{0.5}$ (g) Plot of ratio $T_n / \big(|S_n|^{0.5} \log^2(|S_n|)\big)$.

Figure 3-5: Results of iMDP on a stochastic LQR problem. Figure 3-5(a) shows the convergence of approximated cost-to-go to the optimal analytical cost-to-go over iterations. Anytime solutions are compared to the analytical optimal solution after 200 and 600 iterations in Figs. 3-5(b)-3-5(c). Mean and 1-σ interval of the error $||J_n - J^*||_{S_n}$ are shown in 3-5(d) using 50 trials. The corresponding mean and standard deviation of the error $||J_n - J^*||_{S_n}$ are depicted on a log-log plot in Fig. 3-5(e). In Fig. 3-5(f), we plot the ratio of $||J_n - J^*||_{S_n}$ to $(\log(|S_n|)/|S_n|)^{0.5}$ to show the convergence rate of J_n to J^*. Figure 3-5(g) shows the ratio of running time per iteration T_n to $|S_n|^{0.5} \log^2(|S_n|)$. Ratios in Figs. 3-5(f)-3-5(g) are averaged over 50 trials.

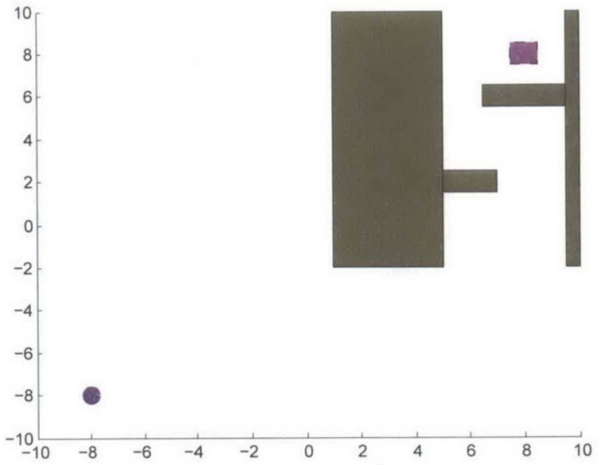


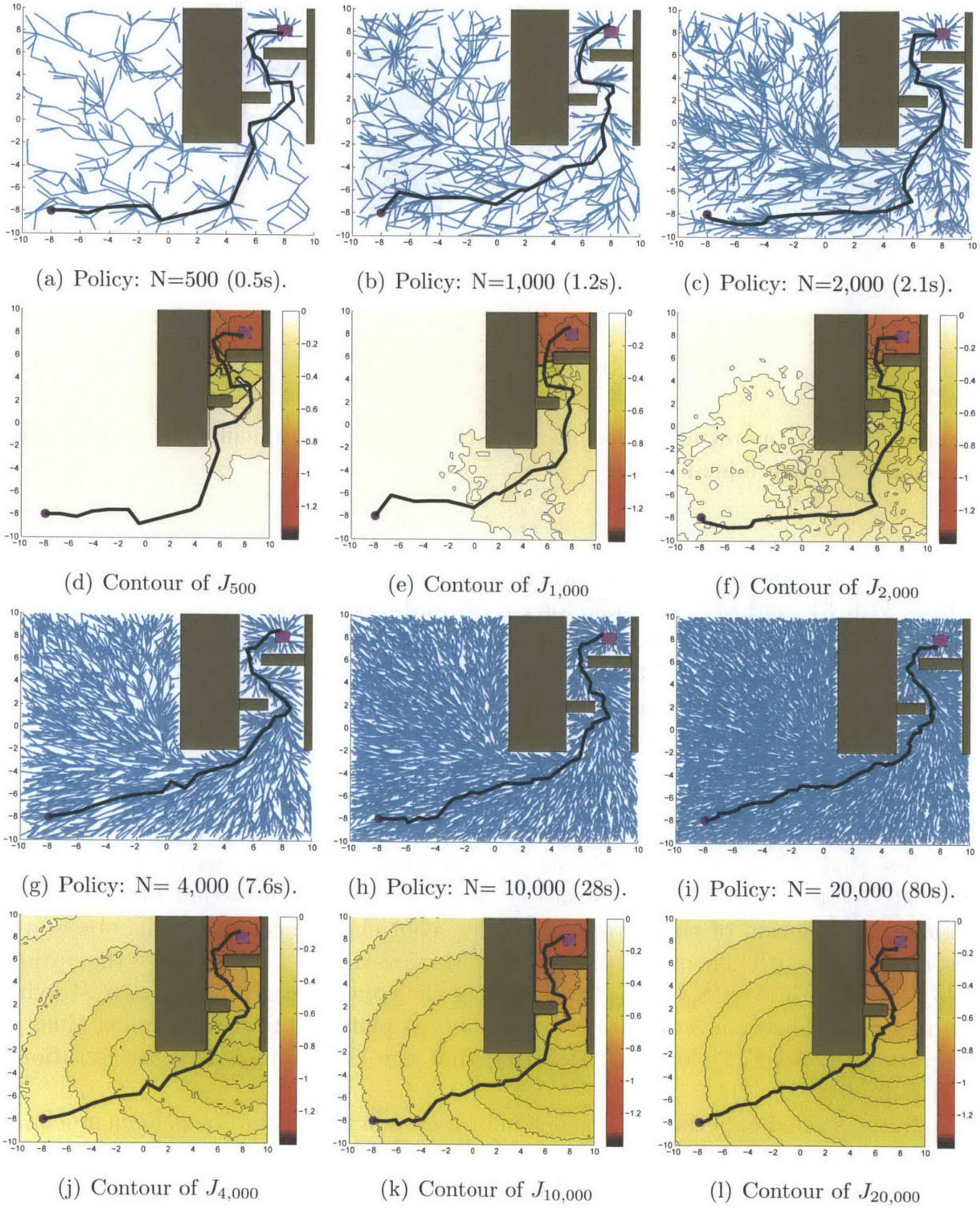
Figure 3-6: An operating environment for the second experiment. The system starts at (-8,8) to reach a goal at (8,8).

on the state space $S = [-6, 6]$ where τ is the first hitting time to the boundary $\partial S = \{-6, 6\}$, and $h(z) = 414.55$ for $z \in \partial S$ and 0 otherwise.

Solving the associate HJB equation, we have that the optimal cost-to-go from $x(0) = z$ is $10.39z^2 + 40.51$, and the optimal control policy is $u(t) = -0.5714x(t)$. Since the cost-rate function is bounded on S and Hölder continuous with exponent 1.0, we use $\rho = 0.5$. In addition, we choose $\theta = 0.5$, and $\varsigma = 0.99$ in the procedure `ComputeHoldingTime`.

Figures 3-5(a)-3-5(c) show the convergence of approximated cost-to-go, anytime controls and trajectory to the optimal analytical counterparts over iterations. We observe that in Fig. 3-5(d), both the mean and variance of cost-to-go error decreases quickly to zero. The log-log plot in Fig. 3-5(e) clearly indicates that both mean and standard deviation of the error $||J_n - J^*||_{S_n}$ continue to decrease. This observation is consistent with Theorems 3.4.5-3.4.6. Moreover, Fig. 3-5(f) shows the ratio of $||J_n - J^*||_{S_n}$ to $(\log(|S_n|)/|S_n|)^{0.5}$ indicating the convergence rate of J_n to J^*, which agrees with Theorem 3.4.4. Finally, Fig. 3-5(g) plots the ratio of running time per iteration T_n to $|S_n|^{0.5}\log(|S_n|)$ asserting that the time complexity per iteration is $O\big(|S_n|^{0.5}\log^2(|S_n|)\big)$.

In the second experiment, we controlled a system with two-dimensional stochastic single integrator dynamics to a goal region with free ending time in a cluttered environment. The dynamics is given by $dx(t) = u(t)dt + Fdw(t)$ where $x(t) \in \mathbb{R}^2$, $u(t) \in \mathbb{R}^2$, and $F = \begin{bmatrix} 0.26 & 0 \\ 0 & 0.26 \end{bmatrix}$. The objective function is discounted with $\alpha = 0.95$. The system pays zero cost for each action it takes and pays a cost of -1 when reaching the goal region $\mathcal{X}_{goal}$ (see Fig. 3-6). The maximum velocity in each direction of the system is one. The system stops when it collides with obstacles. We show how the system reaches the goal in the upper right corner and avoids obstacles with different anytime controls. Anytime control policies after up-to 2,000 iterations in Figs. 3-7(a)-3-7(c), which were obtained within 2.1 seconds, indicate that iMDP



(a) Policy: N=500 (0.5s).

(b) Policy: N=1,000 (1.2s).

(c) Policy: N=2,000 (2.1s).

(d) Contour of J_{500}

(e) Contour of $J_{1,000}$

(f) Contour of $J_{2,000}$

(g) Policy: N= 4,000 (7.6s).

(h) Policy: N= 10,000 (28s).

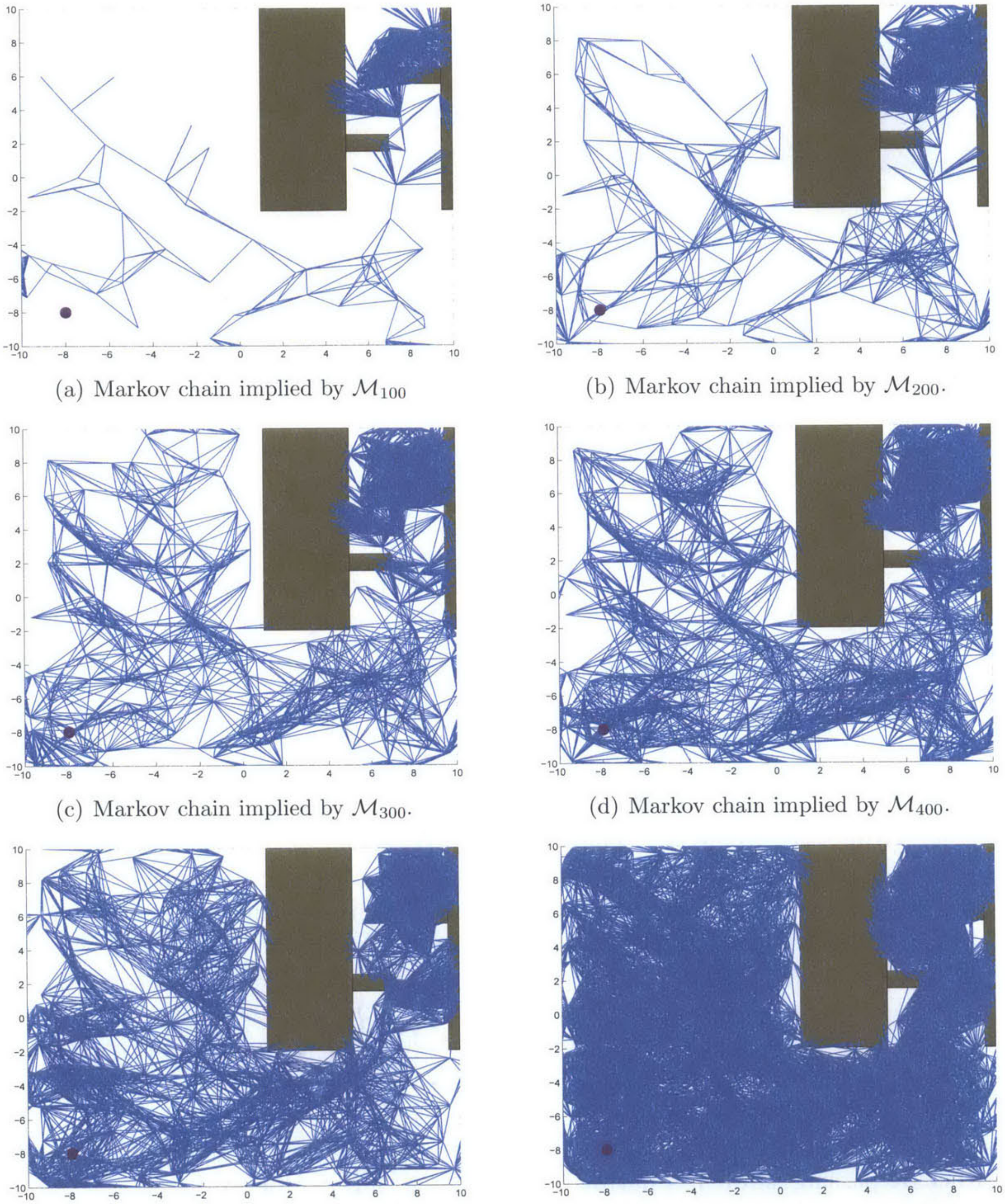
(i) Policy: N= 20,000 (80s).

(j) Contour of $J_{4,000}$

(k) Contour of $J_{10,000}$

(l) Contour of $J_{20,000}$

Figure 3-7: A system with stochastic single integrator dynamics in a cluttered environment. With appropriate cost structure assigned to the goal and obstacle regions, the system reaches the goal in the upper right corner and avoids obstacles. The standard deviation of noise in x and y directions is 0.26. The maximum velocity is one. Anytime control policies and corresponding contours of approximated cost-to-go as shown in Figs. 3-7(a)-3-7(l) indicate that iMDP quickly explores the state space and refines control policies over time.



(a) Markov chain implied by $\mathcal{M}_{100}$

(b) Markov chain implied by $\mathcal{M}_{200}$.

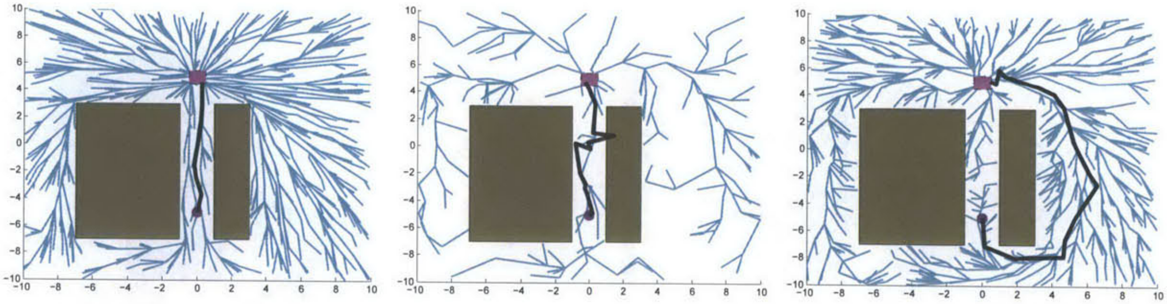
(c) Markov chain implied by $\mathcal{M}_{300}$.

(d) Markov chain implied by $\mathcal{M}_{400}$.

(e) Markov chain implied by $\mathcal{M}_{500}$.

(f) Markov chain implied by $\mathcal{M}_{1000}$.

Figure 3-8: Markov chains over iterations. The structures of these Markov chains are indeed random graphs that are asymptotically almost-surely connected to cover the state space S.

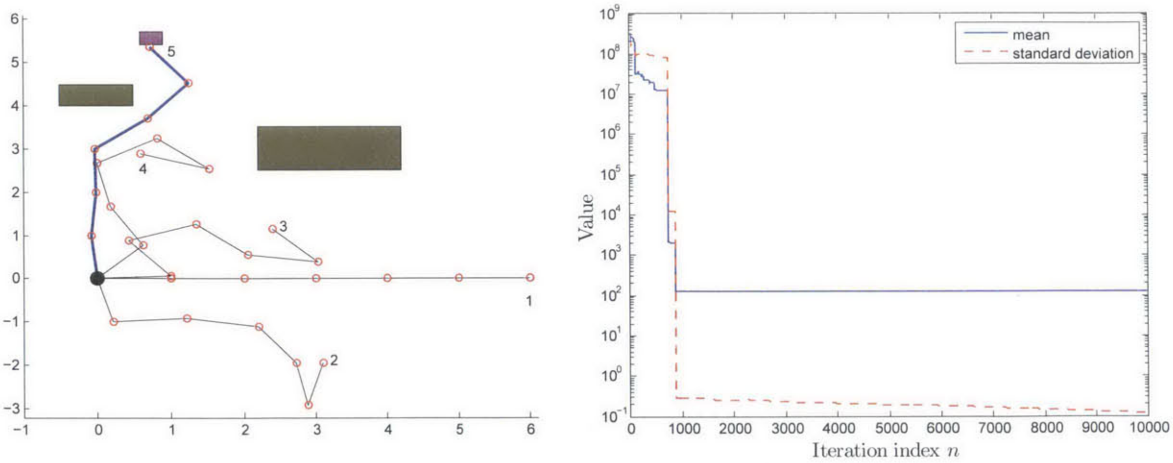


(a) Noise-free: N= 1,000 (1.2s). (b) Stochastic: N= 300(0.4s). (c) Stochastic: N= 1,000 (1.1s).

Figure 3-9: Performance against different process noise magnitude. The system starts from (0,-5) to reach the goal. In Fig. 3-9(a), the environment is noise-free. In Figs. 3-9(b)-3-9(c), standard deviation of noise in x and y directions is 0.37. In the latter, the system first discovers an unsafe route that is prone to collisions and discovers a safer route after a few seconds. (In Fig. 3-9(b), we temporarily let the system continue even after collision to observe the entire trajectory.)

quickly explores the state space and refines control policies over time. Corresponding contours of cost value functions are shown in Figs. 3-7(d)-3-7(f) further illustrate the refinement and convergence of cost value functions to the original optimal cost-to-go over time. We observe that the performance is suitable for real-time control. Furthermore, anytime control policies and cost value functions after up-to 20,000 iterations are shown in Figs. 3-7(g)-3-7(i) and Figs. 3-7(j)-3-7(l) respectively. We note that the control policies seem to converge faster than cost value functions over iterations. The phenomenon is due to the fact that cost value functions J_n are the estimates of the optimal cost-to-go J^*. Thus, when $J_n(z) - J^*(z)$ is constant for all $z \in S_n$, updated controls after a Bellman update are close to their optimal values. Thus, the phenomenon favors the use of the iMDP algorithm in real-time applications where only a small number of iterations are executed. In addition, in Fig. 3-8, we show the Markov chains that are induced by the stored controls over iterations. As we can see, the structures of these Markov chains are indeed random graphs that are asymptotically almost-surely connected to cover the state space S. This observation agrees with the claim provided in Theorem 3.4.1.

In the third experiment, we tested the effect of process noise magnitude on the solution trajectories. In Figs. 3-9(a)-3-9(c), the system wants to arrive at a goal area either by passing through a narrow corridor or detouring around the two blocks. In Fig. 3-9(a), when the dynamics is noise-free (by setting a small dispersion matrix), the iMDP algorithm quickly determines to follow a narrow corridor. In contrast, when the environment affects the dynamics of the system (Figs. 3-9(b)-3-9(c)), the iMDP algorithm decides to detour to have a safer route. This experiment demonstrates the benefit of iMDP in handling process noise compared to RRT-like algorithms [51, 52]. We emphasize that although iMDP spends slightly more time on computation per iteration, iMDP provides feedback policies rather than open-loop policies; thus, replanning is not crucial in iMDP.



(a) Trajectory snapshots after 3000 iterations (15.8s).

(b) Mean and standard deviation of cost values $J_n(x_0)$.

Figure 3-10: Results of a 6D manipulator example. The system is modeled as a single integrator with states representing angles between segments and the horizontal line. Control magnitude is bounded by 0.3. The standard deviation of noise at each joint is 0.032 rad. In Fig. 3-10(a), the manipulator is controlled to reach a goal with the final upright position. In Fig. 3-10(b), the mean and standard deviation of the computed cost values for the initial position are plotted using 50 trials.

In the forth experiment, we examined the performance of the iMDP algorithm for high dimensional systems such as a manipulator with six degrees of freedom. The manipulator is modeled as a single integrator where states represents angles between segments and the horizontal line. Formally, the dynamics is given by $dx(t) = u(t)dt + Fdw(t)$ where $x(t) \in \mathbb{R}^6$ with each component in $[0, 2\pi]$ and $u(t) \in \mathbb{R}^6$. The maximum control magnitude for all joints is 0.3. The dispersion matrix F is such that the standard deviation of noise at each joint is 0.032 rad. The manipulator is controlled to reach a goal with the final upright position in minimum time. In Fig. 3-10(a), we show a resulting trajectory after 3000 iterations computed in 15.8 seconds. In addition, we show the mean and standard deviation of the computed cost values for the initial position using 50 trials in Fig. 3-10(b). As shown in the plots, the solution converges quickly after about 1000 iterations. These results highlight the suitability of the iMDP algorithm to compute feedback policies for complex high dimensional systems in stochastic environments.

3.6 Proofs

In this section, we provide the detailed proofs of theorems presented in Section 3.4.

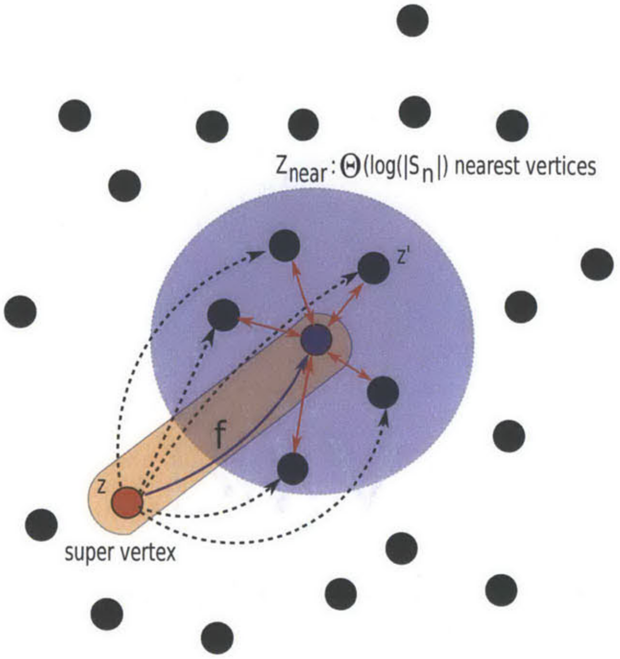


Figure 3-11: An illustration for Lemma 3.6.1. We continue the example in Fig. 3-2. We enlarge each vertex z of $\overrightarrow{G}_n$ to become a "super vertex" $\big(z, z + f(z, \mu_n(z))\Delta t_n\big)$ so that the Euclidean distance between two super vertices $\big(z, z+ f(z, \mu_n(z))\Delta t_n\big)$ and $\big(z', z' + f(z', \mu_n(z'))\Delta t_n\big)$ is defined as the Euclidean distance of $z + f(z, \mu_n(z))\Delta t_n$ and z'. The super vertex is connected to $\Theta(\log(|S_n|))$-nearest vertices using this new distance definition.

3.6.1 Proof of Theorem 3.4.1

Given $\mathcal{M}_n$ and the best stored controls μ_n returned by the iMDP algorithm, we define a directed graph $\overrightarrow{G}_n$ having S_n as its vertex set, and its edges represent transition probabilities under the best stored controls. In particular, for each vertex $z \in S_n \backslash \partial S_n$, we form a directed edge from z to each vertex in the support of $P_n(\cdot \mid z, \mu_n(z))$ that is returned from the procedure `ComputeTranProb`. Vertices from ∂S_n connect to themselves.

We enlarge each vertex z of $\overrightarrow{G}_n$ to become a "super vertex" $\big(z, z+ f(z, \mu_n(z))\Delta t_n\big)$ so that the Euclidean distance between two super vertices $\big(z, z+ f(z, \mu_n(z))\Delta t_n\big)$ and $\big(z', z'+ f(z', \mu_n(z'))\Delta t_n\big)$ is defined as the Euclidean distance of $z+ f(z, \mu_n(z))\Delta t_n$ and z'. Since the support size of $P_n(\cdot \mid z, \mu_n(z))$ is $\Theta(\log(|S_n|))$, $\overrightarrow{G}_n$ is a random directed kNN graph where $k = \Theta(\log(|S_n|)$. Figure 3.6.1 shows an illustration of a super vertex and its nearest neighbors. By Theorem 2.3.4, $\overrightarrow{G}_n$ is connected asymptotically almost surely.

The Markov chain $\{\xi_i^n; i \in \mathbb{N}\}$, which is formed by following the transition probabilities P_n using the best control $\mu_n(z)$ for each state $z \in S_n$, has states that move along edges of $\overrightarrow{G}_n$. When $\overrightarrow{G}_n$ is connected, starting from any non-absorbing vertex in $S_n \backslash \partial S_n$, we can reach an absorbing state in ∂S_n. Therefore, $\{\xi_i^n; i \in \mathbb{N}\}$ is an absorbing Markov chain asymptotically almost surely as n approaches ∞. ▮

3.6.2 Proof of Lemma 3.4.2

For each $n \in \mathbb{N}$, divide the state space S into grid cells with side length $\frac{1}{2}\gamma_r \frac{\log |S_n|}{|S_n|^{1/d_x}}$ as follows. Let $\mathbb{Z}$ denote the set of integers. Define the grid cell $i \in \mathbb{Z}^{d_x}$ as

$$W_n(i) := i \left(\frac{\gamma_r}{2} \frac{\log |S_n|}{|S_n|}\right)^{1/d_x} + \left[-\frac{1}{4}\gamma_r \left(\frac{\log |S_n|}{|S_n|}\right)^{1/d_x}, \frac{1}{4}\gamma_r \left(\frac{\log |S_n|}{|S_n|}\right)^{1/d_x}\right]^{d_x},$$

where $[-a, a]^{d_x}$ denotes the d_x-dimensional cube with side length $2\,a$ centered at the origin. Hence, the expression above translates the d_x-dimensional cube with side length $(1/2)\,\gamma_r(\log |S_n|/|S_n|)^{1/d_x}$ to the point with coordinates $i\,\frac{\gamma_r}{2}(\log n/n)^{1/d_x}$.

Let Q_n denote the indices of set of all cells that lie completely inside the state space S, i.e., $Q_n = \{i \in \mathbb{Z}^d : W_n(i) \subseteq S\}$. Clearly, Q_n is finite since S is bounded. Let ∂Q_n denote the set of all grid cells that intersect the boundary of S, i.e.,

$$\partial Q_n = \{i \in \mathbb{Z}^d : W_n(i) \cap \partial S \neq \emptyset\}.$$

We claim for all large n, all grid cells in Q_n contain one vertex of S_n, and all grid cells in ∂Q_n contain one vertex from ∂S_n. First, let us show that each cell in Q_n contains at least one vertex. Given an event A, let A^c denote its complement. Let $A_{n,k}$ denote the event that the cell $W_n(k)$, where $k \in Q_n$ contains a vertex from S_n, and let A_n denote the event that all grid cells in Q_n contain a vertex in S_n. Then, for all $k \in Q_n$,

$$\begin{aligned}\mathbb{P}\left(A_{n,k}^c\right) = \left(1 - \frac{(\gamma_r/2)^{d_x}}{m(S)} \frac{\log |S_n|}{|S_n|}\right)^{|S_n|} &\leq \exp\left(-\left((\gamma_r/2)^{d_x}/m(S)\right) \log |S_n|\right) \\ &= |S_n|^{-(\gamma_r/2)^{d_x}/m(S)},\end{aligned}$$

where $m(S)$ denotes the Lebesgue measure assigned to S. Then,

$$\begin{aligned}\mathbb{P}(A_n^c) = \mathbb{P}\left(\left(\bigcap\nolimits_{k\in Q_n} A_{n,k}\right)^c\right) &= \mathbb{P}\left(\bigcup\nolimits_{k \in Q_n} A_{n,k}^c\right) \\ &\leq \sum\nolimits_{k\in Q_n} \mathbb{P}\left(A_{n,k}^c\right) = |Q_n|\,|S_n|^{-(\gamma_r/2)^{d_x}/m(S)},\end{aligned}$$

where the first inequality follows from the union bound and $|Q_n|$ denotes the cardinality of the set Q_n. By calculating the maximum number of cubes that can fit into S, we can bound $|Q_n|$:

$$|Q_n| \leq \frac{m(S)}{(\gamma_r/2)^{d_x} \frac{\log |S_n|}{|S_n|}} = \frac{m(S)}{(\gamma_r/2)^{d_x}} \frac{|S_n|}{\log |S_n|}.$$

Note that by construction, we have $|S_n| = \Theta(n)$. Thus,

$$\begin{aligned}\mathbb{P}\left(A_n^c\right) &\le \frac{m(S)}{(\gamma_r/2)^{d_x}} \frac{|S_n|}{\log|S_n|} |S_n|^{-(\gamma_r/2)^{d_x}/m(S)} = \frac{m(S)}{(\gamma_r/2)^{d_x}} \frac{1}{\log|S_n|} |S_n|^{1-(\gamma_r/2)^{d_x}/m(S)} \\ &\le \frac{m(S)}{(\gamma_r/2)^{d_x}} |S_n|^{1-(\gamma_r/2)^{d_x}/m(S)},\end{aligned}$$

which is summable for all $\gamma_r > 2\left(2\,m(S)\right)^{1/d_x}$. Hence, by the Borel-Cantelli lemma, the probability that A_n^c occurs infinitely often is zero, which implies that the probability that A_n occurs for all large n is one, i.e., $\mathbb{P}(\liminf_{n\to\infty} A_n) = 1$.

Similarly, each grid cell in ∂Q_n can be shown to contain at least one vertex from ∂S_n for all large n, with probability one. This implies each grid cell in both sets Q_n and ∂Q_n contain one vertex of S_n and ∂S_n, respectively, for all large n, with probability one. Hence the following event happens with probability one:

$$\zeta_n = \max_{z \in S_n} \min_{z' \in S_n} ||z' - z||_2 = O((\log|S_n|/|S_n|)^{1/d_x}).$$

∎

3.6.3 Proof of Lemma 3.4.3

We show that each state that is added to the approximating MDPs is updated infinitely often. That is, for any $z \in S_n$, the set of all iterations in which the procedure `Update` is applied on z is unbounded. Indeed, let us denote $\zeta_n(z) = \min_{z' \in S_n} ||z' - z||_2$. From Lemma 3.4.2, $\lim_{n\to\infty} \zeta_n(z) = 0$ happens almost surely. Therefore, with probability one, there are infinitely many n such that $\zeta_n(z) < \zeta_{n-1}(z)$. In other words, with probability one, we can find infinitely many z_{new} at Line 13 of Algorithm 1 such that z is updated. For those n, the holding time at z is recomputed as $\Delta t_n(z) = \gamma_t \left(\frac{\log|S_n|}{|S_n|}\right)^{\theta\varsigma\rho/d_x}$ at Line 1 of Algorithm 2. Thus, the following event happens with probability one:

$$\lim_{n\to\infty} \Delta t_n(z) = 0,$$

which satisfies the first condition of local consistency in Eq. (3.7).

The other conditions of local consistency in Eqs. (3.8)-(3.10) are satisfied immediately by the way that the transition probabilities are computed (see the description of the procedure `ComputeTranProb` given in Section 3.3). Hence, the MDP sequence $\{\mathcal{M}_n\}_{n=0}^{\infty}$ and holding times $\{\Delta t_n\}_{n=0}^{\infty}$ are locally consistent for large n with probability one. ∎

3.6.4 Proof of Theorem 3.4.5

To highlight the idea of the entire proof, we first prove the convergence under synchronous value iterations before presenting the convergence under asynchronous value iterations. As we will see, the shrinking rate of holding times plays a crucial role in the convergence proof. The outline of the proof is as follows.

S1: Convergence under synchronous value iterations: In Algorithm 1, we take $L_n \geq 1$ and $K_n = |S_n| - 1$. In other words, in each iteration, we perform synchronous value iterations. Moreover, we assume that we are able to solve the Bellman equation (Eq. (3.15)) exactly. We show that J_n converges uniformly to J^* almost surely in this setting.

S2: Convergence under asynchronous value iterations: When $K_n = \Theta(|S_n|^\theta) < |S_n|$, we only update a subset of S_n in each of L_n passes. We show that J_n still converges uniformly to J^* almost surely in this new setting.

In the following discussion and next sections, we need to compare functions on different domains S_n. To ease the discussion and simplify the notation, we adopt the following interpolation convention. Given $X \subset Y$ and $J : X \to \mathbb{R}$, we interpolate J to $\overline{J}$ on the entire domain Y via nearest neighbor value:

$$\forall y \in Y : \quad \overline{J}(y) = J(z) \text{ where } z = \text{argmin}_{z' \in X} ||z' - y||.$$

To compare $J : X \to \mathbb{R}$ and $J' : Y \to \mathbb{R}$ where $X, Y \subset S$, we define the sup-norm:

$$||J - J'||_\infty = ||\overline{J} - \overline{J'}||_\infty,$$

where $\overline{J}$ and $\overline{J'}$ are interpolations of J and J' from the domains X and Y to the entire domain S respectively. In particular, given $J_n : S_n \to \mathbb{R}$, and $J : S \to \mathbb{R}$, then $||J_n - J||_{S_n} \leq ||J_n - J||_\infty$. Thus, if $||J_n - J||_\infty$ approaches 0 when n approaches ∞, so does $||J_n - J||_{S_n}$. Hence, we will work with the (new) sup-norm $|| \cdot ||_\infty$ instead of $|| \cdot ||_{S_n}$ in the proofs of Theorems 3.4.5-3.4.6. The triangle inequality also holds for any functions J, J', J'' defined on subsets of S with respect to the above sup-norm:

$$||J - J'||_\infty \leq ||J - J''||_\infty + ||J'' - J'||_\infty.$$

Let $B(X)$ denote a set of all real-valued bounded functions over a domain X. For $S_n \subset S_{n'}$ when $n < n'$, a function J in $B(S_n)$ also belongs to $B(S_{n'})$, meaning that we can interpolate J on S_n to a function J' on $S_{n'}$. In particular, we say that J in $B(S_n)$ also belongs to $B(S)$.

Lastly, due to random sampling, S_n is a random set, and therefore functions J_n and J_n^* defined on S_n are random variables. In the following discussion, inequalities hold surely without further explanation when it is clear from the context, and inequalities hold almost surely if they are followed by "w.p.1".

S1: Convergence under synchronous value iterations

In this step, we first set $L_n \geq 1$ and $K_n = |S_n| - 1$ in Algorithm 1. Thus, for all $z \in S_n$, the holding time $\Delta t_n(z)$ equals $\gamma_t \left(\frac{\log |S_n|}{|S_n|}\right)^{\theta\varsigma\rho/d_x}$ and is denoted as Δt_n. We consider the MDP $\mathcal{M}_n = (S_n, U, P_n, G_n, H_n)$ at n^{th} iteration and define the following operator $T_n : B(S_n) \to B(S_n)$ that transforms every $J \in B(S_n)$ after a Bellman update as:

$$T_n J(z) = \min_{v \in U} \{G_n(z, v) + \alpha^{\Delta t_n} \mathbb{E}_{P_n} [J(y)|z, v]\}, \quad \forall z \in S_n, \tag{3.16}$$

assuming that we can solve the minimization on the RHS of Eq. (3.16) exactly. For each $k \geq 2$, operators T_n^k are defined recursively as $T_n^k = T_n T_n^{k-1}$ and $T_n^1 = T_n$. When we apply T_n on $J \in B(S_k)$ where $k < n$, J is interpolated to S_n before applying T_n. Thus, in Algorithms 1-2, we implement the next update

$$J_n = T_n^{L_n} J_{n-1}.$$

Lemma 3.6.1 (Contraction mapping, see [33]) *Given T_n as defined above, T_n is a contraction mapping, i.e. for any J and J' in $B(S_n)$, the following inequality happens surely:*

$$||T_n J - T_n J'||_\infty \leq \alpha^{\Delta t_n} ||J - J'||_\infty.$$

Moreover, $J_n^* = T_n J_n^*$.

Using Lemma 3.6.1:

$$\begin{aligned} ||J_n^* - J_n||_\infty = ||T_n^{L_n} J_n^* - T_n^{L_n} J_{n-1}||_\infty &\leq \alpha^{L_n \Delta t_n} ||J_n^* - J_{n-1}||_\infty \\ &\leq \alpha^{\Delta t_n} (||J_n^* - J_{n-1}^*||_\infty + ||J_{n-1}^* - J_{n-1}||_\infty), \end{aligned}$$

where the second inequality follows from the triangle inequality, and $L_n \geq 1, \alpha \in (0, 1)$.

Thus, by iterating over n, for any $N \geq 1$ and $n > N$, we have:

$$||J_n^* - J_n||_\infty \leq A_n + \alpha^{\Delta t_n + \Delta t_{n-1} ... + \Delta t_{N+1}} ||J_N^* - J_N||_\infty, \tag{3.17}$$

where A_n are defined recursively:

$$A_n = \alpha^{\Delta t_n} (||J_n^* - J_{n-1}^*||_\infty + A_{n-1}), \quad \forall n > N + 1, \tag{3.18}$$

$$A_{N+1} = \alpha^{\Delta t_{N+1}} ||J_{N+1}^* - J_N^*||_\infty. \tag{3.19}$$

Note that for any $N \geq 1$:

$$\lim_{n \to \infty} \Delta t_n + \Delta t_{n-1} ... + \Delta t_{N+1} = \infty,$$

as holding times $\Delta t_n = \gamma_t \left(\frac{\log |S_n|}{|S_n|} \right)^{\theta\varsigma\rho/d_x}$ in the procedure `ComputeHoldingTime`. Therefore,

$$\lim_{n \to \infty} \alpha^{\Delta t_n + ... + \Delta t_{N+1}} ||J_N^* - J_N||_\infty = 0.$$

By Theorem 3.4.4, the following event happens with probability 1 (w.p.1):

$$\lim_{n \to \infty} ||J_n^* - J^*||_\infty = 0,$$

hence,

$$\lim_{n\to\infty} ||J_n^* - J_{n-1}^*||_\infty = 0 \text{ w.p.1.}$$

Thus, for any fixed $\epsilon > 0$, we can choose N large enough such that:

$$||J_n^* - J_{n-1}^*||_\infty^{1-\varsigma} < \epsilon \text{ w.p.1 for all } n > N, \text{ and} \tag{3.20}$$

$$\alpha^{\Delta t_n + \ldots + \Delta t_{N+1}} ||J_N^* - J_N||_\infty < \epsilon \text{ surely}, \tag{3.21}$$

where $\varsigma \in (0, 1)$ is the constant defined in the procedure `ComputeHoldingTime`.

Now, for all $n > N$, we rearrange Eqs. (3.18)-(3.19) to have

$$A_n \leq \epsilon B_n \text{ w.p.1},$$

where

$$B_n = \alpha^{\Delta t_n}(||J_n^* - J_{n-1}^*||_\infty^\varsigma + B_{n-1}), \quad \forall n > N + 1,$$
$$B_{N+1} = \alpha^{\Delta t_{N+1}} ||J_{N+1}^* - J_N^*||_\infty^\varsigma.$$

We can see that for $n > N + 1$:

$$B_n = \alpha^{\Delta t_n}(||J_n^* - J_{n-1}^*||_\infty^\varsigma + B_{n-1}) < \quad \epsilon^{\varsigma/(1-\varsigma)} + B_{n-1} \text{ w.p.1}, \tag{3.22}$$

$$B_{N+1} = \alpha^{\Delta t_{N+1}} ||J_{N+1}^* - J_N^*||_\infty^\varsigma < \quad \epsilon^{\varsigma/(1-\varsigma)} \text{ w.p.1.} \tag{3.23}$$

We now prove that almost surely, B_n is bounded for all $n \geq N$ w.p.1:

Lemma 3.6.2 *B_n is bounded for all $n \geq N$ w.p.1.*

Proof Indeed, we derive the conditions so that $B_{n-1} < B_n$ as follows:

$$\begin{aligned}
& B_{n-1} < B_n \\
\Leftrightarrow \;\; & B_{n-1} < \alpha^{\Delta t_n}(||J_n^* - J_{n-1}^*||_\infty^\varsigma + B_{n-1}) \\
\Leftrightarrow \;\; & B_{n-1} < \frac{\alpha^{\Delta t_n} ||J_n^* - J_{n-1}^*||_\infty^\varsigma}{1 - \alpha^{\Delta t_n}} \\
\Rightarrow \;\; & B_{n-1} < \mathcal{K} \frac{\alpha^{\gamma_t \left(\frac{\log |S_n|}{|S_n|}\right)^{\theta\varsigma\rho/d_x}} \left(\frac{\log |S_n|}{|S_n|}\right)^{\varsigma\rho/d_x}}{1 - \alpha^{\gamma_t \left(\frac{\log |S_n|}{|S_n|}\right)^{\theta\varsigma\rho/d_x}}} \text{ w.p.1.}
\end{aligned}$$

The last inequality is due to Theorem 3.4.4 and $|S_n| = \Theta(n)$, $|S_{n-1}| = \Theta(n - 1)$:

$$||J_n^* - J_{n-1}^*||_\infty = O((\log |S_{n-1}|/|S_{n-1}|)^{\rho/d_x}) < \mathcal{K} \left(\frac{\log |S_n|}{|S_n|}\right)^{\rho/d_x} \text{ w.p.1},$$

for large n where $\mathcal{K}$ is some finite constant. Let $\beta = \alpha^{\gamma_t} \in (0, 1)$. For large n, $\frac{\log |S_n|}{|S_n|}$

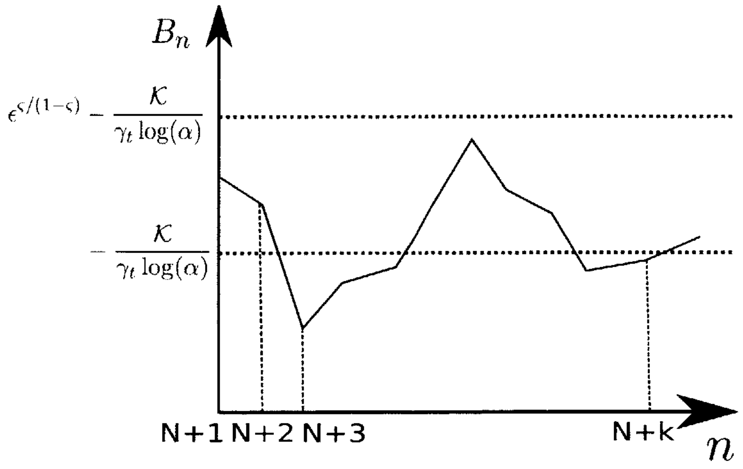


Figure 3-12: A realization of the random sequence B_n. We have B_{N+1} less than $\epsilon^{\varsigma/(1-\varsigma)}$ w.p.1. For n larger than $N+1$, when $B_{n-1} \geq -\frac{\mathcal{K}}{\gamma_t \log(\alpha)}$ w.p.1, the sequence is non-increasing w.p.1, i.e. $B_{n-1} \geq B_n$ w.p.1. Conversely, when the sequence is increasing, i.e. $B_{n-1} < B_n$, we have $B_{n-1} < -\frac{\mathcal{K}}{\gamma_t \log(\alpha)}$ w.p.1, and the increment is less than $\epsilon^{\varsigma/(1-\varsigma)}$. Hence, the random sequence B_n is bounded by $\epsilon^{\varsigma/(1-\varsigma)} - \frac{\mathcal{K}}{\gamma_t \log(\alpha)}$ w.p.1.

are in $(0,1)$ and $\theta \in (0,1]$. Let us define

$$x_n = \left(\frac{\log |S_n|}{|S_n|}\right)^{\theta\varsigma\rho/d_x}, \text{ and } \quad y_n = \left(\frac{\log |S_n|}{|S_n|}\right)^{\varsigma\rho/d_x}.$$

Then, $x_n \geq y_n > 0$. The above condition is simplified to

$$B_{n-1} < \mathcal{K}\frac{\beta^{x_n} y_n}{1-\beta^{x_n}} \leq \mathcal{K}\frac{\beta^{x_n} x_n}{1-\beta^{x_n}}, \text{ w.p.1.}$$

Consider the function $r : [0,\infty) \to \mathbb{R}$ such that $r(x) = \frac{\beta^x x}{1-\beta^x}$, we can verify that $r(x)$ is non-increasing and is bounded by $r(0) = -1/\log(\beta)$. Therefore:

$$B_{n-1} < B_n \quad \Rightarrow \quad B_{n-1} < -\frac{\mathcal{K}}{\log(\beta)} = -\frac{\mathcal{K}}{\gamma_t \log(\alpha)} \quad \text{w.p.1.} \tag{3.24}$$

Or conversely,

$$B_{n-1} \geq -\frac{\mathcal{K}}{\gamma_t \log(\alpha)} \quad \text{w.p.1} \quad \Rightarrow \quad B_{n-1} \geq B_n \quad \text{w.p.1.} \tag{3.25}$$

The above discussion characterizes the random sequence B_n. In particular, Fig. 3-12 shows a possible realization of the random sequence B_n for $n > N$. As shown visually in this plot, B_{N+1} is less than $\epsilon^{\varsigma/(1-\varsigma)}$ w.p.1 and thus is less than $\epsilon^{\varsigma/(1-\varsigma)} - \frac{\mathcal{K}}{\gamma_t \log(\alpha)}$ w.p.1. For $n > N+1$, assume that we have already shown that B_{n-1} is bounded from above by $\epsilon^{\varsigma/(1-\varsigma)} - \frac{\mathcal{K}}{\gamma_t \log(\alpha)}$ w.p.1. When $B_{n-1} \geq -\frac{\mathcal{K}}{\gamma_t \log(\alpha)}$ w.p.1, the sequence is non-increasing w.p.1. Conversely, when the sequence is increasing, i.e. $B_{n-1} < B_n$, we assert that $B_{n-1} < -\frac{\mathcal{K}}{\gamma_t \log(\alpha)}$ w.p.1 due to Eq. (3.24), and the increment is less than $\epsilon^{\varsigma/(1-\varsigma)}$ due to Eq. (3.22). In both cases, we conclude that B_n is also bounded by $\epsilon^{\varsigma/(1-\varsigma)} - \frac{\mathcal{K}}{\gamma_t \log(\alpha)}$ w.p.1. Hence, from Eqs. (3.22)-(3.25), we infer that B_n is bounded w.p.1 for all $n > N$:

$$B_n < \epsilon^{\varsigma/(1-\varsigma)} - \frac{\mathcal{K}}{\gamma_t \log(\alpha)} \text{ w.p.1.}$$

∎

Thus, from Lemma 3.6.2, for all $n > N$:

$$A_n \leq \epsilon B_n < \epsilon\Big(\epsilon^{\varsigma/(1-\varsigma)} - \frac{\mathcal{K}}{\gamma_t \log(\alpha)}\Big) \text{ w.p.1.} \tag{3.26}$$

Combining Eqs. (3.17),(3.21), and (3.26), we conclude that for any $\epsilon > 0$, there exists $N \geq 1$ such that for all $n > N$, we have

$$||J_n^* - J_n||_\infty < \epsilon\Big(\epsilon^{\varsigma/(1-\varsigma)} - \frac{\mathcal{K}}{\gamma_t \log(\alpha)} + 1\Big) \text{ w.p.1.}$$

Therefore,

$$\lim_{n\to\infty} ||J_n^* - J_n||_\infty = 0 \text{ w.p.1.}$$

Combining with Theorem 3.4.4:

$$\lim_{n\to\infty} ||J_n^* - J^*||_\infty = 0 \text{ w.p.1,}$$

we obtain

$$\lim_{n\to\infty} ||J_n - J^*||_\infty = 0 \text{ w.p.1.}$$

In the above analysis, the shrinking rate $\left(\frac{\log|S_n|}{|S_n|}\right)^{\theta\varsigma\rho/d_x}$ of holding times plays an important role to construct an upper bound of the sequence B_n. This rate must be slower than the convergence rate $\left(\frac{\log|S_n|}{|S_n|}\right)^{\rho/d_x}$ of J_n^* to J^* so that the function $r(x)$ is bounded, enabling the convergence of cost value functions J_n to the optimal cost-to-go J^*. Remarkably, we have accomplished this convergence by carefully selecting the range $(0, 1)$ of the parameter ς. The role of the parameter θ in this convergence will be clear in Step S2. Lastly, we note that if we are able to obtain a faster convergence rate of J_n^* to J^*, we can have faster shrinking rate for holding times.

S2: Convergence under asynchronous value iterations

When $1 \leq L_n$ and $K_n = \Theta(|S_n|^\theta) < |S_n|$, we first claim the following result:

Lemma 3.6.3 *Consider any increasing sequence $\{n_k\}_{k=0}^\infty$ as a subset of $\mathbb{N}$ such that*

$n_0 = 0$ and $k \leq |S_{n_k}| \leq k^{1/\theta}$. For $J \in B(S)$, we define:

$$A\big(\{n_j\}_{j=0}^k\big) = \alpha^{\Delta t_{n_k}+\Delta t_{n_{k-1}}+\ldots+\Delta t_{n_1}} ||J^*_{n_1} - J||_\infty + \alpha^{\Delta t_{n_k}+\Delta t_{n_{k-1}}+\ldots+\Delta t_{n_2}} ||J^*_{n_2} - J^*_{n_1}||_\infty + \ldots + \alpha^{\Delta t_{n_k}} ||J^*_{n_k} - J^*_{n_{k-1}}||_\infty.$$

The following event happens with probability one:

$$\lim_{k\to\infty} A\big(\{n_j\}_{j=0}^k\big) = 0.$$

Proof We rewrite $A\big(\{n_j\}_{j=0}^k\big) = A_{n_k}$ where A_{n_k} are defined recursively:

$$A_{n_k} = \alpha^{\Delta t_{n_k}}(||J^*_{n_k} - J^*_{n_{k-1}}||_\infty + A_{n_{k-1}}), \quad \forall k > K, \tag{3.27}$$

$$A_{n_K} = A\big(\{n_j\}_{j=0}^K\big), \quad \forall K \geq 1. \tag{3.28}$$

We note that

$$\begin{aligned}
&\Delta t_{n_k} + \Delta t_{n_{k-1}} + \ldots + \Delta t_{n_K} \\
&= \gamma_t \left(\frac{\log|S_{n_k}|}{|S_{n_k}|}\right)^{\theta\varsigma\rho/d_x} + \gamma_t \left(\frac{\log|S_{n_{k-1}}|}{|S_{n_{k-1}}|}\right)^{\theta\varsigma\rho/d_x} + \ldots + \gamma_t \left(\frac{\log|S_{n_K}|}{|S_{n_K}|}\right)^{\theta\varsigma\rho/d_x} \\
&\geq \gamma_t \left(\frac{1}{|S_{n_k}|}\right)^{\theta\varsigma\rho/d_x} + \gamma_t \left(\frac{1}{|S_{n_{k-1}}|}\right)^{\theta\varsigma\rho/d_x} + \ldots + \gamma_t \left(\frac{1}{|S_{n_K}|}\right)^{\theta\varsigma\rho/d_x} \\
&\geq \gamma_t \frac{1}{k^{\varsigma\rho/d_x}} + \gamma_t \frac{1}{(k-1)^{\varsigma\rho/d_x}} + \ldots + \gamma_t \frac{1}{(K)^{\varsigma\rho/d_x}} \geq \gamma_t (\frac{1}{k} + \frac{1}{k-1} + \ldots + \frac{1}{K}),
\end{aligned}$$

where the second inequality uses the given fact that $|S_{n_k}| \leq k^{1/\theta}$. Therefore, for any $K \geq 1$:

$$\lim_{k\to\infty} \alpha^{\Delta t_{n_k}+\Delta t_{n_{k-1}}\ldots+\Delta t_{n_K}} = 0.$$

We choose a constant $\varrho > 1$ such that $\varrho\varsigma < 1$. For any fixed $\epsilon > 0$, we can choose K large enough such that:

$$||J^*_{n_k} - J^*_{n_{k-1}}||_\infty^{1-\varrho\varsigma} < \epsilon \text{ w.p.1 for all } k > K. \tag{3.29}$$

For all $k > K$, we can write

$$A_{n_k} \leq \epsilon B_{n_k} + \alpha^{\Delta t_{n_k}+\ldots+\Delta t_{n_{K+1}}} A\big(\{n_j\}_{j=0}^K\big).$$

where

$$\begin{aligned}
B_{n_k} &= \alpha^{\Delta t_{n_k}}(||J^*_{n_k} - J^*_{n_{k-1}}||_\infty^{\varrho\varsigma} + B_{n_{k-1}}), \quad \forall k > K, \\
B_{n_K} &= 0.
\end{aligned}$$

Furthermore, we can choose K' sufficiently large such that $K' \geq K$ and for all $k > K'$:

$$\alpha^{\Delta t_{n_k}+\ldots+\Delta t_{n_{K+1}}} A\big(\{n_j\}_{j=0}^K\big) \leq \epsilon.$$

We obtain:

$$A_{n_k} \leq \epsilon B_{n_k} + \epsilon, \quad \forall k > K' \geq K \geq 1.$$

We can also see that for $k > K$:

$$B_{n_k} = \alpha^{\Delta t_{n_k}}(||J^*_{n_k} - J^*_{n_{k-1}}||_\infty^{\varrho\varsigma} + B_{n_{k-1}}) < \quad \epsilon^{\varrho\varsigma/(1-\varrho\varsigma)} + B_{n_{k-1}} \text{ w.p.1.} \tag{3.30}$$

Similar to Step S1, we characterize the random sequence B_{n_k} as follows:

$$\begin{aligned} & B_{n_{k-1}} < B_{n_k} \\ \Leftrightarrow \quad & B_{n_{k-1}} < \frac{\alpha^{\Delta t_{n_k}} ||J^*_{n_k} - J^*_{n_{k-1}}||_\infty^{\varrho\varsigma}}{1 - \alpha^{\Delta t_{n_k}}} \\ \Rightarrow \quad & B_{n_{k-1}} < \mathcal{K} \frac{\alpha^{\gamma_t \left(\frac{\log |S_{n_k}|}{|S_{n_k}|}\right)^{\theta\varsigma\rho/d_x}} \left(\frac{\log |S_{n_{k-1}}|}{|S_{n_{k-1}}|}\right)^{\varrho\varsigma\rho/d_x}}{1 - \alpha^{\gamma_t \left(\frac{\log |S_{n_k}|}{|S_{n_k}|}\right)^{\theta\varsigma\rho/d_x}}} \text{ w.p.1.} \end{aligned}$$

Let $\beta = \alpha^{\gamma_t} \in (0,1)$. We define:

$$x_k = \left(\frac{\log |S_{n_k}|}{|S_{n_k}|}\right)^{\theta\varsigma\rho/d_x}, \text{ and } \quad y_k = \left(\frac{\log |S_{n_{k-1}}|}{|S_{n_{k-1}}|}\right)^{\varrho\varsigma\rho/d_x}.$$

We note that $\frac{\log x}{x}$ is a decreasing function for positive x. Since $|S_{n_{k-1}}| \geq k-1$ and $|S_{n_k}| \leq k^{1/\theta}$, we have the following inequalities:

$$x_k \geq \left(\frac{(\frac{\log k}{\theta})^\theta}{k}\right)^{\varsigma\rho/d_x}, \quad y_k \leq \left(\frac{(\log(k-1))^\varrho}{(k-1)^\varrho}\right)^{\varsigma\rho/d_x}.$$

Since $\theta \in (0,1]$ and $\varrho > 1$, we can find a finite constant $\mathcal{K}_1$ such that $y_k < \mathcal{K}_1 x_k$ for large k. Thus, the above condition leads to

$$B_{n_{k-1}} < \mathcal{K} \frac{\beta^{x_k} y_k}{1 - \beta^{x_k}} < \mathcal{K}\mathcal{K}_1 \frac{\beta^{x_k} x_k}{1 - \beta^{x_k}}, \text{ w.p.1.}$$

Therefore:

$$B_{n_{k-1}} < B_{n_k} \quad \Rightarrow \quad B_{n_{k-1}} < -\frac{\mathcal{K}\mathcal{K}_1}{\log(\beta)} = -\frac{\mathcal{K}\mathcal{K}_1}{\gamma_t \log(\alpha)} \quad \text{w.p.1.}$$

Or conversely,

$$B_{n_{k-1}} \geq -\frac{\mathcal{K}\mathcal{K}_1}{\gamma_t \log(\alpha)} \quad \text{w.p.1} \quad \Rightarrow \quad B_{n-1} \geq B_n \quad \text{w.p.1.}$$

Arguing similarly to Step S1, we infer that for all $k > K' \geq K \geq 1$:

$$B_{n_k} < \epsilon^{\varrho\varsigma/(1-\varrho\varsigma)} - \frac{\mathcal{K}\mathcal{K}_1}{\gamma_\iota \log(\alpha)} \text{ w.p.1.}$$

Thus, for any $\epsilon > 0$, we can find $K' \geq 1$ such that for all $k > K'$:

$$A_{n_k} \leq \epsilon B_{n_k} + \epsilon < \epsilon\Big(\epsilon^{\varrho\varsigma/(1-\varrho\varsigma)} - \frac{\mathcal{K}\mathcal{K}_1}{\gamma_\iota \log(\alpha)} + 1\Big) \text{ w.p.1.}$$

We conclude that

$$\lim_{k\to\infty} A\big(\{n_j\}_{j=0}^k\big) = 0. \text{ w.p.1.}$$

∎

Returning to the main proof, we use the tilde notation to indicate asynchronous operations to differentiate with our synchronous operations in Step S1. We will also assume that $L_n = 1$ for all n to simplify the following notations. The proof for general $L_n \geq 1$ is exactly the same. We define the following (asynchronous) mappings $\widetilde{T}_n : B(S_n) \to B(S_n)$ as the restricted mappings of T_n on D_n, a non-empty random subset of S_n, such that for all $J \in B(S_n)$:

$$\widetilde{T}_n J(z) = \min_{v\in U}\Big\{G_n(z,v) + \alpha^{\Delta t_n}\mathbb{E}_{P_n}\big[J(y)|z,v\big]\Big\}, \quad \forall z \in D_n \subset S_n, \tag{3.31}$$

$$\widetilde{T}_n J(z) = J(z), \quad \forall z \in S_n \backslash D_n. \tag{3.32}$$

We require that

$$\cap_{n=1}^{\infty} \cup_{k=n}^{\infty} D_k = S. \tag{3.33}$$

In other words, every state in S are sampled infinitely often. We can see that in Algorithm 1, if the set Z_{update} is assigned to D_n in every iteration (Line 13), the sequence $\{D_n\}_{n=1}^{\infty}$ has the above property, and $|D_n| = \Theta(|S_n|^\theta) < |S_n|$.

Starting from any $\widetilde{J}_0 \in B(S_0)$, we perform the following asynchronous iteration

$$\widetilde{J}_{n+1} = \widetilde{T}_{n+1}\widetilde{J}_n, \quad \forall n \geq 0. \tag{3.34}$$

Consider the following sequence $\{m_k\}_{k=0}^{\infty}$ such that $m_0 = 0$ and for all $k \geq 0$, from m_k to $m_{k+1} - 1$, all states in $S_{m_{k+1}-1}$ are chosen to be updated at least once, and a subset of states in $S_{m_{k+1}-1}$ is chosen to be updated exactly once. We observe that as the size of S_n increases linearly with n, if we schedule states in $D_n \subset S_n$ to be updated in a round-robin manner, we have $k \leq S_{m_k} \leq k^{1/\theta}$. When D_n is chosen as shown in Algorithm 1, with high probability, $k \leq S_{m_k} \leq k^{1/\theta}$. However, we will assume that the event $k \leq S_{m_k} \leq k^{1/\theta}$ happens surely because we can always schedule a fraction of D_n to be updated in a round-robin manner.

We define W_n as the set of increasing sub-sequences of the sequence $\{0, 1, ..., n\}$ such that each sub-sequence contains $\{m_j\}_{j=0}^k$ where $m_k \leq n < m_{k+1}$:

$$W_n = \Big\{ \{i_j\}_{j=0}^T \mid \{m_j\}_{j=0}^k \subset \{i_j\}_{j=0}^T \subset \{0, 1, ..., n\} \wedge T \geq 2 \wedge m_k \leq n < m_{k+1} \Big\}.$$

Clearly, if $\{i_j\}_{j=0}^T \in W_n$, we have $i_0 = 0$. For each $\{i_j\}_{j=0}^T \in W_n$, we define

$$A\big(\{i_j\}_{j=0}^T\big) = \alpha^{\Delta t_{i_T} + \Delta t_{i_{T-1}} + ... + \Delta t_{i_1}} ||J_{i_1}^* - \widetilde{J}_0||_\infty + \alpha^{\Delta t_{i_T} + \Delta t_{i_{T-1}} + ... + \Delta t_{i_2}} ||J_{i_2}^* - J_{i_1}^*||_\infty \\ + ... + \alpha^{\Delta t_{i_T}} ||J_{i_T}^* - J_{i_{T-1}}^*||_\infty.$$

We will prove by induction that

$$\forall z \in D_n \Rightarrow |\widetilde{J}_n(z) - J_n^*(z)| \leq \max_{\{i_j\}_{j=0}^T \in W_n} A\big(\{i_j\}_{j=0}^T\big). \tag{3.35}$$

When $n = 1$, the only sub-sequence is $\{i_j\}_{j=0}^T = \{0, 1\} \in W_1$. It is clear that for $z \in D_1$, due to the contraction property of T_1:

$$|J_1^*(z) - \widetilde{J}_1(z)| \leq \max_{\{i_j\}_{j=0}^T \in W_1} A\big(\{i_j\}_{j=0}^T\big) = \alpha^{\Delta t_1} ||J_1^* - \widetilde{J}_0||_\infty.$$

Assuming that Eq. (3.35) holds up-to $n = m_k$, we need to prove that the equation also holds for those $n \in (m_k, m_{k+1})$ and $n = m_{k+1}$. Indeed, let us assume that Eq. (3.35) holds for some $n \in [m_k, m_{k+1} - 1)$. Denote $n_z \leq n$ as the index of the most recent update of z. For $z \in D_n$, we compute new values for z in $\widetilde{J}_{n+1}$, and by the contraction property of T_{n+1}, it follows that

$$\begin{aligned} |\widetilde{J}_{n+1}(z) - J_{n+1}^*(z)| &\leq \alpha^{\Delta t_{n+1}} ||J_{n+1}^* - \widetilde{J}_n||_\infty \\ &= \alpha^{\Delta t_{n+1}} \max_{z \in S_{n+1}} |J_{n+1}^*(z) - \widetilde{J}_n(z)| \\ &= \alpha^{\Delta t_{n+1}} \max_{z \in S_{n+1}} |J_{n+1}^*(z) - \widetilde{J}_{n_z}(z)| \\ &\leq \alpha^{\Delta t_{n+1}} \max_{z \in S_{n+1}} \big(|J_{n_z}^*(z) - \widetilde{J}_{n_z}(z)| + ||J_{n+1}^* - J_{n_z}^*||_\infty\big) \\ &\leq \max_{z \in S_{n+1}} \Big(\alpha^{\Delta t_{n+1}} \max_{\{i_j\}_{j=0}^T \in W_{n_z}} A\big(\{i_j\}_{j=0}^T\big) + \alpha^{\Delta t_{n+1}} ||J_{n+1}^* - J_{n_z}^*||_\infty\Big) \\ &= \max_{\{i_j\}_{j=0}^T \in W_{n+1}} A\big(\{i_j\}_{j=0}^T\big). \end{aligned}$$

The last equality is due to $n + 1 \leq m_{k+1} - 1$, and $\{m_j\}_{j=0}^k \subset \{\{i_j\}_{j=0}^T, n + 1\} \subset \{0, 1, ..., n + 1\}$ for any $\{i_j\}_{j=0}^T \in W_{n_z}$. Therefore, Eq. (3.35) holds for all $n \in$

$(m_k, m_{k+1}-1]$. When $n = m_{k+1}-1$, we also have the above relation for all $z \in D_{n+1}$:

$$
\begin{aligned}
|\widetilde{J}_{n+1}(z) - J^*_{n+1}(z)| &\le \max_{z\in S_{n+1}} \Big(\alpha^{\Delta t_{n+1}} \max_{\{i_j\}_{j=0}^T \in W_{n_z}} A\big(\{i_j\}_{j=0}^T\big) + \alpha^{\Delta t_{n+1}} ||J^*_{n+1} - J^*_{n_z}||_\infty\Big) \\
&= \max_{\{i_j\}_{j=0}^T \in W_{n+1}} A\big(\{i_j\}_{j=0}^T\big).
\end{aligned}
$$

The last equality is due to $n+1 = m_{k+1}$ and thus $\{m_j\}_{j=0}^{k+1} \subset \{\{i_j\}_{j=0}^T, n+1\} \subset \{0, 1, ..., n+1\}$ for any $\{i_j\}_{j=0}^T \in W_{n_z}$. Therefore, Eq. (3.35) also holds for $n = m_{k+1}$ and this completes the induction.

We see that all $\{i_j\}_{j=0}^T \in W_n$, we have $j \le i_j \le m_j$, and thus $j \le S_{i_j} \le j^{1/\theta}$. By Lemma 3.6.3,

$$
\lim_{n\to\infty} A\big(\{i_j\}_{j=0}^T \in W_n\big) = 0 \text{ w.p.1.}
$$

Therefore,

$$
\lim_{n\to\infty} \sup_{z\in D_n} |\widetilde{J}_n(z) - J^*_n(z)| = 0 \text{ w.p.1.}
$$

Since all states are updated infinitely often, and J^*_n converges uniformly to J^* with probability one, we conclude that:

$$
\lim_{n\to\infty} ||\widetilde{J}_n - J^*_n||_\infty = 0 \text{ w.p.1.}
$$

and

$$
\lim_{n\to\infty} ||\widetilde{J}_n - J^*||_\infty = 0 \text{ w.p.1.}
$$

In both Steps S1 and S2, we have $\lim_{n\to\infty} ||J_n - J^*_n||_\infty = 0$ w.p.1 [8], therefore μ_n converges to μ^*_n pointwise w.p.1 as μ_n and μ^*_n are induced from Bellman updates based on J_n and J^*_n respectively. Hence, the sequence of policies $\{\mu_n\}_{n=0}^\infty$ has each policy μ_n as an ϵ_n-optimal policy for the MDP $\mathcal{M}_n$ such that $\lim_{n\to\infty} \epsilon_n = 0$. By Theorem 3.2.3, we conclude that

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

∎

3.6.5 Proof of Theorem 3.4.6

We fix an initial starting state $x(0) = z$. In Theorem 3.4.5, starting from an initial state $x(0) = z$, we construct a sequence of Markov chains $\{\xi^n_i; i \in \mathbb{N}\}_{n=1}^\infty$ under minimizing control sequences $\{u^n_i; i \in \mathbb{N}\}_{n=1}^\infty$. By convention, we denote the associated interpolated continuous time trajectories and control processes as $\{\xi^n(t); t \in \mathbb{R}\}_{n=1}^\infty$ and $\{u^n(t); t \in \mathbb{R}\}_{n=1}^\infty$ respectively. By Theorem 3.2.2, $\{\xi^n(t); t \in \mathbb{R}\}_{n=1}^\infty$ converges in distribution to an optimal trajectory $\{x^*(t); t \in \mathbb{R}\}$ under an optimal control process $\{u^*(t); t \in \mathbb{R}\}$ with probability one. In other words, $(\xi^n(\cdot), u^n(\cdot)) \xrightarrow{d} (x^*(\cdot), u^*(\cdot))$

[8] The tilde notion is dropped at this point.

w.p.1. We will show that this result can hold even when the Bellman equation is not solved exactly at each iteration.

In this theorem, we solve the Bellman equation (Eq. (3.15)) by sampling uniformly in U to form a control set U_n such that $\lim_{n\to\infty}|U_n| = \infty$. Let us denote the resulting Markov chains and control sequences due to this modification as $\{\overline{\xi}_i^n; i \in \mathbb{N}\}_{n=1}^{\infty}$ and $\{\overline{u}_i^n; i \in \mathbb{N}\}_{n=1}^{\infty}$ with associated continuous time interpolations $\{\overline{\xi}^n(t); t \in \mathbb{R}\}_{n=1}^{\infty}$ and $\{\overline{u}^n(t); t \in \mathbb{R}\}_{n=1}^{\infty}$. In this case, randomness is due to both state and control sampling. We will prove that there exists minimizing control sequences $\{u_i^n; i \in \mathbb{N}\}_{n=1}^{\infty}$ and the induced sequence of Markov chains $\{\xi_i^n; i \in \mathbb{N}\}_{n=1}^{\infty}$ in Theorem 3.4.5 such that

$$(\overline{\xi}^n(\cdot) - \xi^n(\cdot), \overline{u}^n(\cdot) - u^n(\cdot)) \xrightarrow{P} (0,0), \tag{3.36}$$

where $(0,0)$ denotes a pair of zero processes. To prove Eq. (3.36), we first prove the following lemmas. In the following analysis, we assume that the Bellman update (Eq. (3.15)) has minima in a neighborhood of the positive Lebesgue measure. We also assume additional continuity of cost functions for discrete MDPs.

Lemma 3.6.4 *Let us consider the sequence of approximating MDPs $\{\mathcal{M}_n\}_{n=0}^{\infty}$. For each n and a state $z \in S_n$, let v_n^* be an optimal control minimizing the Bellman update, which is referred to as an optimal control from z:*

$$\begin{aligned} v_n^* \in V_n^* &= argmin_{v\in U}\{G_n(z,v) + \alpha^{\Delta t_n(z)}\mathbb{E}_{P_n}\left[J_{n-1}(y)|z,v\right]\}, \\ J_n(z,v_n^*) &= J_n^*(z) = G_n(z,v_n^*) + \alpha^{\Delta t_n(z)}\mathbb{E}_{P_n}\left[J_{n-1}(y)|z,v_n^*\right], \quad \forall v_n^* \in V_n^*. \end{aligned}$$

Let $\overline{v}_n$ be the best control in a sampled control set U_n from z:

$$\begin{aligned} \overline{v}_n &= argmin_{v\in U_n}\{G_n(z,v) + \alpha^{\Delta t_n(z)}\mathbb{E}_{P_n}\left[J_{n-1}(y)|z,v\right]\}, \\ J_n(z,\overline{v}_n) &= G_n(z,\overline{v}_n) + \alpha^{\Delta t_n(z)}\mathbb{E}_{P_n}\left[J_{n-1}(y)|z,\overline{v}_n\right]. \end{aligned}$$

Then, when $\lim_{n\to\infty}|U_n| = \infty$, we have $|J_n(z,\overline{v}_n) - J_n^*(z)| \xrightarrow{P} 0$ as n approaches ∞, and there exists a sequence $\{v_n^* \mid v_n^* \in V_n^*\}_{n=0}^{\infty}$ such that $||\overline{v}_n - v_n^*||_2 \xrightarrow{P} 0$.

Proof We assume that for any $\epsilon > 0$, the set $A_\epsilon^n = \{v \in U|\ |J_n(z,v) - J_n^*(z)| \leq \epsilon\}$ has the positive Lebesgue measure. That is, $m(A_\epsilon^n) > 0$ for all $\epsilon > 0$ where m is the Lebesgue measure assigned to U. For any $\epsilon > 0$, we have:

$$\mathbb{P}\big(\{|J_n(z,\overline{v}_n) - J_n^*(z)| \geq \epsilon\}\big) = \big(1 - m(A_\epsilon^n)/m(U)\big)^{|U_n|}.$$

Since $1 - m(A_\epsilon^n)/m(U) \in [0,1)$ and $\lim_{n\to\infty}|U_n| = \infty$, we infer that:

$$\lim_{n\to\infty}\mathbb{P}\big(\{|J_n(z,\overline{v}_n) - J_n^*(z)| \geq \epsilon\}\big) = 0.$$

Hence, we conclude that $|J_n(z,\overline{v}_n) - J_n^*(z)| \xrightarrow{P} 0$ as $n \to \infty$. Under the mild assumption that $J_n(z,v)$ is continuous on U for all $z \in S_n$, there exists a sequence $\{v_n^* \mid v_n^* \in V_n^*\}_{n=0}^{\infty}$ such that $||\overline{v}_n - v_n^*||_2 \xrightarrow{P} 0$ as n approaches ∞. ∎

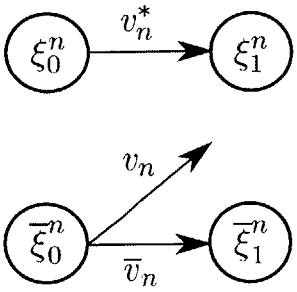


Figure 3-13: An illustration for Lemma 3.6.5. We have $\overline{\xi}_0^n$ converges in probability to ξ_0^n. From ξ_0^n, the optimal control is v_n^* that results in the next random state ξ_1^n. From $\overline{\xi}_0^n$, the optimal control and the best sampled control are v_n and $\overline{v}_n$ respectively. The next random state from $\overline{\xi}_0^n$ due to the control $\overline{v}_n$ is $\overline{\xi}_1^n$.

By Lemma 3.6.4, we conclude that $||J_n - J_n^*||_\infty$ converges to 0 in probability. Thus, J_n returned from the iMDP algorithm when the Bellman update is solved via sampling converges uniformly to J^* in probability. We, however, claim that J_{n,μ_n} still converges pointwise to J^* almost surely in the next discussion.

Lemma 3.6.5 *With the notations in Lemma 3.6.4, consider two states ξ_0^n and $\overline{\xi}_0^n$ such that $||\overline{\xi}_0^n - \xi_0^n||_2 \xrightarrow{p} 0$ as n approaches ∞. Let $\overline{\xi}_1^n$ be the next random state of $\overline{\xi}_0^n$ under the best sampled control $\overline{v}_n$ from $\overline{\xi}_0^n$. Then, there exists a sequence of optimal controls v_n^* from ξ_0^n such that $||\overline{v}_n - v_n^*||_2 \xrightarrow{p} 0$ and $||\overline{\xi}_1^n - \xi_1^n||_2 \xrightarrow{p} 0$ as n approaches ∞, where ξ_1^n is the next random state of ξ_0^n under the optimal control v_n^* from ξ_0^n.*

Proof We have $\overline{v}_n$ as the best sampled control from $\overline{\xi}_0^n$. By Lemma 3.6.4, there exists a sequence of optimal controls v_n from $\overline{\xi}_0^n$ such that $||\overline{v}_n - v_n||_2 \xrightarrow{p} 0$. We assume that the mapping from state space S_n, which is endowed with the usual Euclidean metric, to optimal controls in U is continuous. As $||\overline{\xi}_0^n - \xi_0^n||_2 \xrightarrow{p} 0$, there exists a sequence of optimal controls v_n^* from ξ_0^n such that $||v_n - v_n^*||_2 \xrightarrow{p} 0$. Now, $||\overline{v}_n - v_n||_2 \xrightarrow{p} 0$ and $||v_n - v_n^*||_2 \xrightarrow{p} 0$ lead to $||\overline{v}_n - v_n^*||_2 \xrightarrow{p} 0$ as $n \to \infty$. Figure 3-13 illustrates how $\overline{v}_n, v_n$, and v_n^* relate $\overline{\xi}_1^n$ and ξ_1^n.

Using the probability transition P_n of the MDP $\mathcal{M}_n$ that is locally consistent with the original continuous system, we have:

$$\begin{aligned}
&\mathbb{E}[\xi_1^n \mid \xi_0^n, u_0^n = v_n^*] = \xi_0^n + f(\xi_0^n, v_n^*)\Delta t_n(\xi_0^n) + o(\Delta t_n(\xi_0^n)),\\
&\mathbb{E}[\overline{\xi}_1^n \mid \overline{\xi}_0^n, \overline{u}_0^n = \overline{v}_n] = \overline{\xi}_0^n + f(\overline{\xi}_0^n, \overline{v}_n)\Delta t_n(\overline{\xi}_0^n) + o(\Delta t_n(\overline{\xi}_0^n)),\\
&Cov[\xi_1^n \mid \xi_0^n, u_0^n = v_n^*] = F(\xi_0^n, v_n^*)F(\xi_0^n, v_n^*)^T \Delta t_n(\xi_0^n) + o(\Delta t_n(\xi_0^n)),\\
&Cov[\overline{\xi}_1^n \mid \overline{\xi}_0^n, \overline{u}_0^n = \overline{v}_n] = F(\overline{\xi}_0^n), \overline{v}_n)F(\overline{\xi}_0^n), \overline{v}_n)^T \Delta t_n(\overline{\xi}_0^n)) + o(\Delta t_n(\overline{\xi}_0^n))),
\end{aligned}$$

where $f(\cdot, \cdot)$ is the nominal dynamics, and $F(\cdot, \cdot)F(\cdot, \cdot)^T$ is the diffusion of the original system that are assumed to be continuous almost everywhere. We note that $\Delta t_n(\overline{\xi}_0^n) = \Delta t_n(\xi_0^n) = \gamma_t\big(\log(|S_n|)/|S_n|\big)^{\theta\varsigma\rho/d_x}$ as $\overline{\xi}_0^n$ and ξ_0^n are updated at the n^{th} iteration in this context, and the holding times converge to 0 as n approaches infinity. Therefore, when $||\overline{\xi}_0^n - \xi_0^n||_2 \xrightarrow{p} 0$, $||\overline{v}_n - v_n^*||_2 \xrightarrow{p} 0$, we have:

$$\mathbb{E}[\overline{\xi}_1^n - \xi_1^n \mid \xi_0^n, \overline{\xi}_0^n, u_0^n = v_n^*, \overline{u}_0^n = \overline{v}_n] \xrightarrow{p} 0, \tag{3.37}$$

$$Cov(\overline{\xi}_1^n - \xi_1^n \mid \xi_0^n, \overline{\xi}_0^n, u_0^n = v_n^*, \overline{u}_0^n = \overline{v}_n) \xrightarrow{p} 0. \tag{3.38}$$

Since $\overline{\xi}_1^n$ and ξ_1^n are bounded, the random vector $\mathbb{E}[\overline{\xi}_1^n - \xi_1^n \mid \xi_0^n, \overline{\xi}_0^n, u_0^n = v_n^*, \overline{u}_0^n = \overline{v}_n]$ and random matrix $Cov(\overline{\xi}_1^n - \xi_1^n \mid \xi_0^n, \overline{\xi}_0^n, u_0^n = v_n^*, \overline{u}_0^n = \overline{v}_n)$ are bounded. We recall that if $Y_n \xrightarrow{p} 0$, and hence $Y_n \xrightarrow{d} 0$, when Y_n is bounded for all n, $\lim_{n\to\infty} \mathbb{E}[Y_n] = 0$ and $\lim_{n\to\infty} Cov(Y_n) = 0$. Therefore, Eqs. (3.37)-3.38 imply:

$$\lim_{n\to\infty} \mathbb{E}\Big[\mathbb{E}[\overline{\xi}_1^n - \xi_1^n \mid \xi_0^n, \overline{\xi}_0^n, u_0^n = v_n^*, \overline{u}_0^n = \overline{v}_n]\Big] = 0, \tag{3.39}$$

$$\lim_{n\to\infty} \mathrm{Cov}\Big(\mathbb{E}[\overline{\xi}_1^n - \xi_1^n \mid \xi_0^n, \overline{\xi}_0^n, u_0^n = v_n^*, \overline{u}_0^n = \overline{v}_n]\Big) = 0, \tag{3.40}$$

$$\lim_{n\to\infty} \mathbb{E}\Big[Cov(\overline{\xi}_1^n - \xi_1^n \mid \xi_0^n, \overline{\xi}_0^n, u_0^n = v_n^*, \overline{u}_0^n = \overline{v}_n)\Big] = 0. \tag{3.41}$$

The above outer expectations and covariance are with respect to the randomness of states ξ_0^n, $\overline{\xi}_0^n$ and sampled controls U_n. Using the iterated expectation law for Eq. (3.39), we obtain:

$$\lim_{n\to\infty} \mathbb{E}[\overline{\xi}_1^n - \xi_1^n] = 0.$$

Using the law of total covariance for Eqs. (3.40)-(3.41), we have:

$$\lim_{n\to\infty} Cov[\overline{\xi}_1^n - \xi_1^n] = 0.$$

Since

$$\mathbb{E}[||\overline{\xi}_1^n - \xi_1^n||_2^2] = \mathbb{E}[(\overline{\xi}_1^n - \xi_1^n)^T(\overline{\xi}_1^n - \xi_1^n)] = ||\mathbb{E}[\overline{\xi}_1^n - \xi_1^n)]||_2^2 + tr(Cov[\overline{\xi}_1^n - \xi_1^n]),$$

the above limits together imply:

$$\lim_{n\to\infty} \mathbb{E}[||\overline{\xi}_1^n - \xi_1^n||_2^2] = 0.$$

In other words, $\overline{\xi}_1^n$ converges in 2^{th}-mean to ξ_1^n, which leads to $||\overline{\xi}_1^n - \xi_1^n||_2 \xrightarrow{p} 0$ as n approaches ∞. ∎

Returning to the proof of Eq. (3.36), we know that $\xi_0^n = \overline{\xi}_0^n = z$ as the starting state. From any $y \in S_n$, an optimal control from y is denoted as $v^*(y)$, and the best sampled control from the same state y is denoted as $\overline{v}(y)$.

By Lemma 3.6.5, as $\overline{u}_0^n = \overline{v}(\overline{\xi}_0^n)$, there exists $u_0^n = v^*(\xi_0^n)$ such that $||\overline{u}_0^n - u_0^n||_2 \xrightarrow{p} 0$ and $||\overline{\xi}_1^n - \xi_1^n||_2 \xrightarrow{p} 0$. Let us assume that $(||\overline{u}_{k-1}^n - u_{k-1}^n||_2, ||\overline{\xi}_k^n - \xi_k^n||_2)$ converges in probability to $(0, 0)$ up-to index k. We have $\overline{u}_k^n = \overline{v}(\overline{\xi}_k^n)$. Using Lemma 3.6.5, there exists $u_k^n = v^*(\xi_k^n)$ such that $(||\overline{u}_k^n - u_k^n||_2, ||\overline{\xi}_{k+1}^n - \xi_{k+1}^n||_2) \xrightarrow{p} (0, 0)$. Thus, for any $i \geq 1$, we can construct a minimizing control u_i^n in Theorem 3.4.5 such that

$(||\overline{\xi}_i^n - \xi_i^n||_2, ||\overline{u}_i^n - u_i^n||_2) \xrightarrow{p} (0,0)$ as $n \to \infty$. Hence, Eq. (3.36) follows immediately:

$$(\overline{\xi}^n(\cdot) - \xi^n(\cdot), \overline{u}^n(\cdot) - u^n(\cdot)) \xrightarrow{p} (0,0).$$

We have $(\xi^n(\cdot), u^n(\cdot)) \xrightarrow{d} (x^*(\cdot), u^*(\cdot))$ w.p.1. Thus, by hierarchical convergence of random variables [143], we achieve

$$(\overline{\xi}^n(\cdot), \overline{u}^n(\cdot)) \xrightarrow{d} (x^*(\cdot), u^*(\cdot)) \text{ w.p.1.}$$

Therefore, for all $z \in S_n$:

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

∎

3.6.6 Proof of Theorem 3.4.7

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

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

We assume that optimal policies of the original continuous problem are obtainable. By Theorems 3.4.5-3.4.6, we have:

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

Thus, $\mu_n(y_n)$ 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:

$$||\mu_n(y_n) - \mu^*(z)||_2 \leq ||\mu_n(y_n) - \mu^*(y_n)||_2 + ||\mu^*(y_n) - \mu^*(z)||_2 \leq \epsilon, \ \forall n > N \text{ w.p.1.}$$

Therefore,

$$\lim_{n\to\infty} ||\overline{\mu}_n(z) - \mu^*(z)||_2 = \lim_{n\to\infty} ||\mu_n(y_n) - \mu^*(z)||_2 = 0 \text{ w.p.1.}$$

∎

Chapter 4

Stochastic Control with trajectory Performance Constraints

We now consider a class of stochastic optimal control problems with bounded trajectory performance constraints. The constraints have the same integration structure as the objective functions with different cost rate, terminal cost functions and possibly different discount factors.

Examples of these constraints are trajectory performance requirements such as fuel consumption requirements on autonomous cars, stealthiness requirements for aircraft, and thermal control requirements on spacecraft. The formulation in this chapter enforces these constraints for all sub-trajectories. As a special case, we can *approximately* enforce the probability that a system enters undesirable regions to remain below a certain threshold. We will handle exact probability constraints that are enforced for only initial states in Chapter 5.

In the following, we discuss an extended iMDP algorithm that approximates arbitrarily well an optimal feedback policy of the constrained problem. We show that in the presence of the considered constraints, the sequence of policies returned from the algorithm is both probabilistically sound and asymptotically optimal. Subsequently, we demonstrate the proposed algorithm on motion planning and control problems in the presence of process noise.[1]

4.1 Problem Formulation

We consider a system with the same dynamics (Eq. (3.1)) in Chapter 3 in a bounded state space S:

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

[1] Results in this chapter have been presented in [144].

We want to find a control policy μ to minimize the same objective function:

$$J_\mu(z) = \mathbb{E}\left[\int_0^{T_{\mu,z}} \alpha^t\, g\big(x(t), \mu(x(t))\big)\, dt + \alpha^{T_{\mu,z}} h(x(T_{\mu,z})) \mid x(0) = z\right],$$

where an extra subscript component z in the first exit time $T_{\mu,z}$ emphasizes the dependence of the first exit time on an initial state z.

In addition, we consider trajectory constraints under a policy μ of the form

$$C_\mu(z') \in \Gamma \textit{ for all } z' \in S, \tag{4.1}$$

where

$$C_\mu(z') = \mathbb{E}\left[\int_0^{T_{\mu,z'}} \beta^t\, r\big(x(t), \mu(x(t))\big)\, dt + \beta^{T_{\mu,z'}} k(x(T_{\mu,z'})) \mid x(0) = z'\right], \tag{4.2}$$

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 $[0, 1)$. In other words, the constraints evaluate the distribution of trajectories starting from z' based on criteria encoded by $r(\cdot, \cdot)$ and $k(\cdot)$ until the system first hits the boundary of S. As we specify the constraint for all $z' \in S$, intuitively, the constraints of the form in Eq. (4.1) enforce the value $C_\mu(\cdot)$ for *every sub-trajectory* under the policy μ to be within Γ.

For simplicity, we consider one trajectory constraint in this paper, and handling multiple trajectory constraints is exactly the same. The resulting *optimal cost-to-go function* $J^* : S \to \overline{\mathbb{R}}$ is defined for all $z \in S$ in the next optimization problem:

$$\mathcal{OPT}2: \quad J^*(z) = \inf_{\mu \in \Pi} J_\mu(z) \tag{4.3}$$

$$s/t \qquad C_\mu(z') \in \Gamma,\ \forall z' \in S. \tag{4.4}$$

As in the previous chapter, we call a sampling-based algorithm asymptotically-optimal if the sequence of solutions returned from the algorithm converges to an optimal solution in probability as the number of samples approaches infinity. In addition, 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. Solutions returned from algorithms with the above properties are thus called probabilistically-sound and asymptotically-optimal.

In the next section, we extend the iMDP algorithm to approximate the optimal cost-to-go function and an optimal policy of $\mathcal{OPT}2$ in an anytime fashion so that the returned solutions are both probabilistically-sound and asymptotically-optimal.

4.2 Extended iMDP Algorithm

We approximate the dynamics and cost function on discrete-state MDPs as described in Section 3.2. In particular, the cost-to-go function on an MDP $\mathcal{M}_n$ under a policy

$\mu_n \in \Pi_n$ has the following form:

$$J_{n,\mu_n}(z) = \mathbb{E}_{P_n}\left[\sum_{i=0}^{I_n-1} \alpha^{t_i^n} G_n(\xi_i^n, \mu_n(\xi_i^n)) + \alpha^{t_{I_n}^n} H_n(\xi_{I_n}^n) \;\middle|\; \xi_0^n = z\right].$$

The continuous trajectory constraint is similarly approximated as $C_{n,\mu_n}(z') \in \Gamma$ for all $z' \in S_n$:

$$C_{n,\mu_n}(z') = \mathbb{E}_{P_n}\left[\sum_{i=0}^{I_n-1} \beta^{t_i^n} R_n(\xi_i^n, \mu_n(\xi_i^n)) + \beta^{t_{I_n}^n} K_n(\xi_{I_n}^n) \;\middle|\; \xi_0^n = z'\right], \qquad (4.5)$$

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

Thus, the optimal cost function on $\mathcal{M}_n$, denoted by J_n^*, is defined in the following approximating optimization problem:

$$\mathcal{M\_OPT}2: \; J_n^*(z) = \inf_{\mu_n \in \Pi_n} J_{n,\mu_n}(z) \qquad (4.6)$$

$$s/t \qquad C_{n,\mu_n}(z') \in \Gamma, \; \forall z' \in S_n. \qquad (4.7)$$

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 $||J_{n,\mu_n} - J_n^*||_\infty \leq \epsilon$.

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

The iMDP algorithm is presented in Algorithms 4-6 in which we use the same primitive procedures in Chapter 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. (3.1). 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 constraint value function over S_n respectively.

Initially, an empty MDP model is created. In every main iteration of Algorithm 4, 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* in Chapter 3, 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

Algorithm 4: trajectory constrained iMDP()

```
1  $(n, S_0, J_0, \mu_0, \Delta t_0) \leftarrow (1, \emptyset, \emptyset, \emptyset, \emptyset)$;
2  for $n = 1 \to N$ do
3      $(S_n, J_n, C_n, \mu_n, \Delta t_n) \leftarrow (S_{n-1}, J_{n-1}, C_{n-1}, \mu_{n-1}, \Delta t_{n-1})$;
       // Add a new state to the boundary
4      $z_s \leftarrow$ SampleBoundary();
5      $(S_n, J_n(z_s), C_n(z_s), \mu_n(z_s), \Delta t_n(z_s)) \leftarrow (S_n \cup \{z_s\}, h(z_s), k(z_s), \emptyset, 0)$ ;
       // Add a new state to the interior
6      $z_s \leftarrow$ Sample();
7      $z_{\text{nearest}} \leftarrow$ Nearest($z_s, S_n, 1$);
8      if $(x_{\text{new}}, u_{\text{new}}, \tau) \leftarrow$ ExtendBackwards($z_{\text{nearest}}, z_s, T_0$) then
9          $z_{\text{new}} \leftarrow x_{new}(0)$;
10         $cost = \tau g(z_{\text{new}}, u_{\text{new}}) + \alpha^\tau J_n(z_{\text{nearest}})$;
11         $consValue = \tau r(z_{\text{new}}, u_{\text{new}}) + \beta^\tau C_n(z_{\text{nearest}})$;
           // Discard if constraint value not in $\Gamma$
12         if $consValue \notin \Gamma$ then
13             continue ;
14         $(S_n, J_n(z_{\text{new}}), C_n(z_{\text{new}}), \mu_n(z_{\text{new}}), \Delta t_n(z_{\text{new}})) \leftarrow$
           $(S_n \cup \{z_{\text{new}}\}, cost, consValue, u_{new}, \tau)$ ;
           // Perform $L_n \geq 1$ updates
15         for $i = 1 \to L_n$ do
               // Choose $K_n = \Theta(|S_n|^\theta) < |S_n|$ states
16             $Z_{\text{update}} \leftarrow$ Nearest($z_{\text{new}}, S_n \backslash \partial S_n, K_n$) $\cup \{z_{\text{new}}\}$;
17             for $z \in Z_{\text{update}}$ do
18                 Update($z, S_n, J_n, \mu_n, \Delta t_n$);
```

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 5 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 are 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

Algorithm 5: 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($M_n, z, S_n, \tau$);
3 for $v \in U_n$ do
4     ($Z_{\text{near}}, p_n$) $\leftarrow$ ComputeTranProb($z, v, \tau, S_n$);
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 6: Policy($z \in S, n$)

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

each $v \in U_n$, we construct the support Z_{near} and compute the transition probability $P_n(\cdot \mid z, v)$ consistently over Z_{near} from the procedure ComputeTranProb (Line 4). The cost values and induced constraint values for the state z and controls in U_n are computed at Lines 5-6. We finally choose the best control in U_n that yields the smallest updated cost value and *feasible constraint value* (Line 8). Again, as the current control may be still the best control compared to other controls in U_n, in Algorithm 5, 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.

Finally, for each $n \in \mathbb{N}$, the control policy μ_n is described in Algorithm 6, which is the same as presented in the original version of the iMDP algorithm.

4.3 Analysis

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 4 at the end n iterations. As shown in Section 3.4, 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. (3.1) almost surely. 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^*.

Theorem 4.3.1 *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 $z \in S_n$:

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

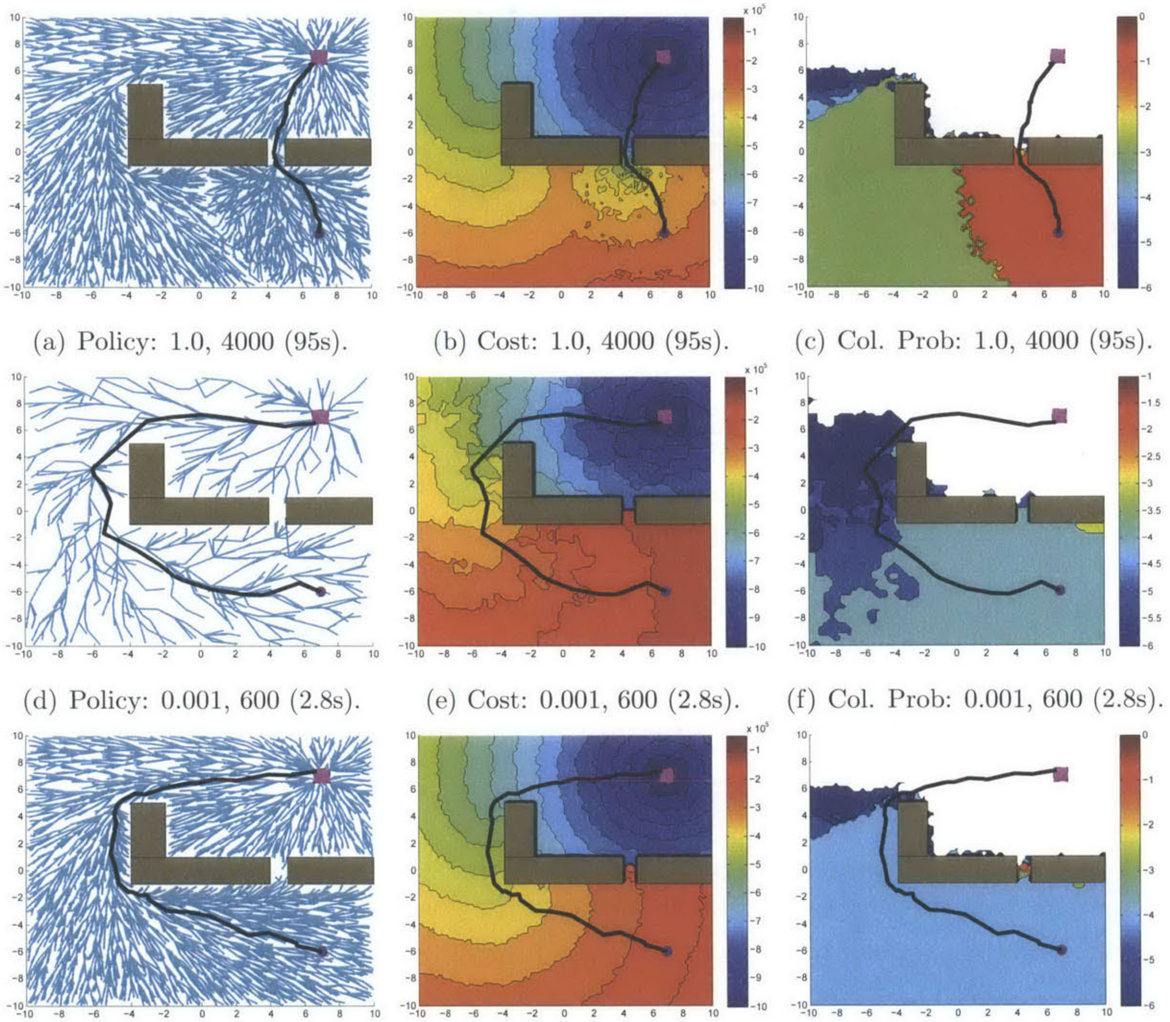
As a corollary, $C_{\mu^*}(z) \in \Gamma$ w.p.1 for all $z \in \cup_{n=0}^{\infty} S_n$. That is, the sequence $\{\mu_n\}_{n=0}^{\infty}$ is probabilistically sound.

The proof of this algorithm follows directly from our analysis in Section 3.4. The almost sure pointwise convergence of J_{n,μ_n} to J^* have been proven in Theorem 3.4.6. 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 $\mu_n(z)$.

4.4 Experiments

We controlled a system with stochastic single integrator dynamics to a goal region with free ending time in a cluttered environment. We consider again the dynamics $dx(t) = u(t)dt + Fdw(t)$ where $x(t) \in \mathbb{R}^2$, $u(t) \in \mathbb{R}^2$, and $F = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}$. 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 a 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 *approximately expresses* the collision probability under the control policy using a large discount factor (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 *collision probabilities*.

We first set the upper value of the collision probability to 1.0, i.e. $\Gamma = (0, 1.0]$. Figures 4-1(a)-4-1(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



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

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

(c) Col. Prob: 1.0, 4000 (95s).

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

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

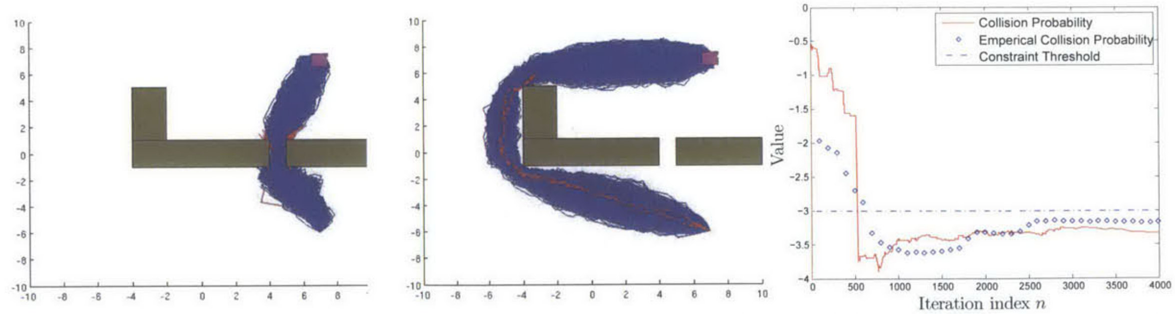
(f) Col. Prob: 0.001, 600 (2.8s).

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

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

(i) Col. Prob: 0.001, 4000 (98s).

Figure 4-1: An example of bounded trajectory performance. 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 *approximately expresses* the probability of collision (with a discount rate $\beta = 0.9999$). Figures 4-1(a)-4-1(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 4-1(d)-4-1(f) and Figures 4-1(g)-4-1(i) show the corresponding plots for the constraint upper bound $0.001(0.1\%)$ after 600 iterations and $4,000$ iterations respectively.



(a) Empirical trajectories for Fig. 4-1(a) (8.2%).

(b) Empirical trajectories for Fig. 4-1(g) (0.07%).

(c) Computed and tested collision probabilities.

Figure 4-2: Bounded trajectory performance results. Figure 4-2(a) shows 10,000 empirical trajectories for the returned policy in Fig. 4-1(a). Collision-free trajectories are plotted in green, and colliding trajectories are plotted in red. The empirical collision probability is 8.2%. Figure 4-2(b) shows 10,000 empirical trajectories for the returned policy in Fig. 3-7(c) with the resulting empirical collision probability 0.07%. When $\Gamma = (0, 0.001]$, in Fig. 4-2(c), 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.

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. 4-1(d)-4-1(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 longer route that stays away from the obstacles. Similarly, Figs. 4-1(g)-4-1(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 the 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 4-2(a) shows 10,000 empirical trajectories for the returned policy in Fig. 4-1(a) when $\Gamma = (0, 1.0]$ where the empirical collision probability is 0.082. Similarly, Fig. 4-2(b) shows 10,000 empirical trajectories for the returned policy in Fig. 4-1(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. 4-2(c). 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.

Chapter 5

Martingale Approach for Risk Management

In this chapter, we consider a class of continuous-time stochastic optimal control problems with risk constraints that are expressed as bounded probabilities of failure for *particular initial states.* For critical applications such as self-driving cars and robotic surgery, regulatory authorities can impose a threshold of failure probability during operation of these systems. Finding control policies that fully respect this type of constraint is important in practice. As opposed to the problem formulation in Chapter 4, the problem formulation in this chapter does not enforce the probability constraints for states along the controlled trajectories. Thus, solutions in this chapter would allow for more aggressive controls. The problem formulation is equivalent to the chance-constrained optimization problem studied in robotics where the probability of safely arriving at a goal from an initial state is required to be above a certain threshold. However, as we discussed in Chapter 1, most previous works in robotics [75, 109–113, 116, 117, 119–121] do not solve the continuous-time problems directly and often modify the problem formulation. As a result, available methods are either computationally intractable or only able to provide approximate but time-inconsistent solutions.

We present here a martingale approach to solve these problems such that obtained control policies are time-consistent with the initial failure-probability threshold. The martingale approach enables us to transform a risk-constrained problem into a stochastic target problem. The martingale represents the consistent variation of risk tolerance that is contingent on available information over time. By sampling in the augmented state space and computing proper boundary values of the reformulated problem, we extend the iMDP algorithm to compute 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. The returned solutions are both probabilistically-sound and asymptotically-optimal.

Compared to available approaches in robotics, the martingale approach fully respects the considered risk constraints for systems with continuous-time dynamics in a time-consistent manner. In addition, the presented algorithm in this chapter constructs incremental solutions without directly deriving the associated HJB equations.

In the following, we provide a formal problem definition and discuss the martingale approach that enables the key transformation. Subsequently, the extended iMDP algorithm, the analysis of the algorithm, and examples on motion planning and control problems are presented.[1]

5.1 Problem Formulation

The notations used to describe the system dynamics and the resulting optimization problem in this chapter follow closely to their counterparts in Chapters 3 and 4. However, as we will see that optimal policies are randomized policies depending on extra random variables, we need to consider a broader class of admissible policies compared to the previous chapters. Thus, we modify our notations slightly to accommodate this purpose. In the following presentation, we will highlight these modifications when necessary.

We consider a system with the same dynamics in Eq. (3.1):

$$dx(t) = f(x(t), u(t))\, dt + F(x(t), u(t))\, dw(t), \forall t \geq 0. \tag{5.1}$$

We recall that $w(\cdot)$ is an $\mathbb{R}^{d_w}$ Brownian motion on a probability space $(\Omega, \mathcal{F}, P)$, and the control process $u(\cdot)$ is admissible with respect to $w(\cdot)$. Let $\mathcal{U}$ be the set of all such control processes.

We define the *first exit time* $T_{u,z} : \mathcal{U} \times S \to [0, +\infty]$ *under a control process* $u(\cdot) \in \mathcal{U}$ starting from $x(0) = z \in S$ as

$$T_{u,z} = \inf \big\{t : x(0) = z,\ x(t) \notin S^o, \text{ and Eq.(3.1)}\big\}. \tag{5.2}$$

The expected cost-to-go function under a control process $u(\cdot)$ is a mapping from S to $\mathbb{R}$ defined as

$$J_u(z) = \mathbb{E}\left[\int_0^{T_{u,z}} \alpha^t\, g\big(x(t), u(t)\big)\, dt + \alpha^{T_{u,z}} h(x(T_{u,z})) \mid x(0) = z\right], \tag{5.3}$$

where the cost rate function $g : S \times U \to \mathbb{R}$ and the terminal cost function $h : S \to \mathbb{R}$ satisfy the same regularity conditions as presented in previous chapters. We remark that the notations $T_{u,z}$ and $J_u(z)$ signify the dependence on a control process $u(\cdot)$ in $\mathcal{U}$ rather than a Markov policy as used in the previous chapters.

Let $\Gamma \subset \partial S$ be a set of failure states, and $\eta \in [0, 1]$ be a threshold for risk tolerance given as a parameter. We consider a risk constraint that is specified for an initial state $x(0) = z$ under a control process $u(\cdot)$ as follows:

$$P_0^z(x(T_{u,z}) \in \Gamma) \leq \eta,$$

where P_t^z denotes the conditional probability at time t given $x(t) = z$. That is, controls that drive the system from time 0 until the first exit time must be consistent with the choice of η and the initial state z at time 0. Intuitively, the constraint

[1] Results in this chapter have been partially published in [145].

enforces that starting from *a given state* z at time $t = 0$, if we execute a control process $u(\cdot)$ for N times, when N is very large, there are at most $N\eta$ executions resulting in failure. Control processes $u(\cdot)$ that satisfy this constraint are called *time-consistent*. To have time-consistent control processes, the risk tolerance along controlled trajectories must vary consistently with the initial choice of risk tolerance η based on available information over time.

Let $\overline{\mathbb{R}}$ be the extended real number set. The *optimal cost-to-go function* $J^* : S \to \overline{\mathbb{R}}$ is defined as follows:

$$\mathcal{OPT}3: \quad J^*(z;\eta) = \inf_{u\in\mathcal{U}} J_u(z) \tag{5.4}$$

$$s/t \qquad P_0^z(x(T_{u,z}) \in \Gamma) \leq \eta \;\; \textit{and Eq. } (5.1). \tag{5.5}$$

In the above notations, the semicolon in $J^*(z;\eta)$ indicates that η is a parameter. A control process $u^*(\cdot)$ is called optimal if $J_{u^*}(z) = J^*(z;\eta)$. For any $\epsilon > 0$, a control process $u(\cdot)$ is called an ϵ-optimal policy if $|J_u(z) - J^*(z;\eta)| \leq \epsilon$. We note that compared to the previous chapters, we consider a larger set of control processes than the set of Markov control processes here. We will restrict again to Markov control processes in the reformulated problem in Section 5.2.

In this chapter, we consider the problem of computing the optimal cost-to-go function J^* and an optimal control process u^* if obtainable for the problem $\mathcal{OPT}3$. We present here a martingale approach to handle the probability constraint and an extended iMDP algorithm that constructs approximate cost-to-go functions and policies that are both probabilistically-sound and asymptotically-optimal.

5.2 Martingale approach

We now discuss the martingale approach that transforms the risk-constrained problem into an equivalent stochastic target problem. The following lemma to diffuse risk constraints is a key tool for our transformation.

5.2.1 Diffusing risk constraints

Lemma 5.2.1 (see [24, 25]) *From $x(0) = z$, a control process $u(\cdot)$ is feasible if and only if there exists an adapted square-integrable (but possibly unbounded) process $c(\cdot) \in \mathbb{R}^{d_w}$ and a martingale $q(\cdot)$ satisfying:*

1. $q(0) = \eta$, *and* $dq(t) = c^T(t)dw(t)$,

2. *For all* t, $q(t) \in [0, 1]$ *a.s.,*

3. $1_\Gamma(x(T_{u,z})) \leq q(T_{u,z})$ *a.s,*

where $1_\Gamma(x) = 1$ if and only if $x \in \Gamma$ and 0 otherwise. The martingale $q(t)$ stands for the level of risk tolerance at time t. We call $c(\cdot)$ a martingale control process.

Proof Assuming that there exists $c(\cdot)$ and $q(\cdot)$ as above, due to the martingale property of $q(\cdot)$, we have:

$$\begin{aligned} P_0^z(x(T_{u,z}) \in \Gamma) &= \mathbb{E}\left[1_\Gamma(x(T_{u,z}))|\mathcal{F}_0\right] \\ &\leq \mathbb{E}\left[q(T_{u,z})|\mathcal{F}_0\right] = q(0) = \eta. \end{aligned}$$

Thus, $u(\cdot)$ is feasible.

Now, let $u(\cdot)$ be a feasible control policy. Set $\eta_0 = P_0^z(x(T_{u,z}) \in \Gamma)$. We note that $\eta_0 \leq \eta$. We define the martingale

$$\overline{q}(t) = \mathbb{E}[1_\Gamma(x(T_{u,z}))|\mathcal{F}_t].$$

Since $\overline{q}(T_{u,z}) \in [0,1]$, we infer that $\overline{q}(t) \in [0,1]$ almost surely. We now set

$$\widehat{q}(t) = \overline{q}(t) + (\eta - \eta_0),$$

then $\widehat{q}(t)$ is a martingale with $\widehat{q}(0) = \overline{q}(0) + (\eta - \eta_0) = \eta_0 + (\eta - \eta_0) = \eta$ and $\widehat{q}(t) \geq 0$ almost surely.

Now, we define $\tau = \inf\{t \in [0, T_{u,z}] \mid \widehat{q}(t) \geq 1\}$, which is a stopping time. Thus,

$$q(t) = \widehat{q}(t)1_{t \leq \tau} + 1_{t > \tau},$$

as a stopped process of the martingale $\widehat{q}(t)$ at τ, is again a martingale with values in [0,1] a.s.

If $\tau < T_{u,z}$, we have

$$1_\Gamma(x(T_{u,z})) \leq 1 = q(T_{u,z}),$$

and if $\tau = T_{u,z}$, we have

$$\begin{aligned} q(T_{u,z}) &= \mathbb{E}[1_\Gamma(x(T_{u,z}))|\mathcal{F}_{T_{u,z}}] + (\eta - \eta_0) \\ &= 1_\Gamma(x(T_{u,z})) + (\eta - \eta_0) \geq 1_\Gamma(x(T_{u,z})). \end{aligned}$$

Hence, $q(\cdot)$ also satisfies that $1_\Gamma(x(T_{u,z})) \leq q(T_{u,z})$.

The control process $c(\cdot)$ exists due to the martingale representation theorem (see Theorem 2.2.5), which yields $dq(t) = c^T(t)dw(t)$. We however note that $c(t)$ is possibly unbounded. We also emphasize that the risk tolerance η becomes the initial value of the martingale $q(\cdot)$. ∎

5.2.2 Stochastic target problem

Using the above lemma, we augment the original system dynamics with the martingale $q(t)$ into the following form:

$$d\begin{bmatrix} x(t) \\ q(t) \end{bmatrix} = \begin{bmatrix} f(x(t), u(t)) \\ 0 \end{bmatrix} dt + \begin{bmatrix} F(x(t), u(t)) \\ c^T(t) \end{bmatrix} dw(t), \tag{5.6}$$

where $(u(\cdot), c(\cdot))$ is the control process of the above dynamics. The initial value of the new state is $(x(0), q(0)) = (z, \eta)$. We will refer to the augmented state space $S \times [0,1]$ as $\overline{S}$ and the augmented control space $U \times \mathbb{R}^{d_w}$ as $\overline{U}$. We also refer to the nominal dynamics and dispersion matrix of Eq. (5.6) as $\overline{f}(x, q, u, c)$ and $\overline{F}(x, q, u, c)$ respectively.

It is well-known that in the following reformulated problem, optimal control processes are Markov controls [24,25,129]. Thus, let us now focus on the set of Markov controls that depend only on the current state, i.e., $(u(t), c(t))$ is a function only of $(x(t), q(t))$, for all $t \geq 0$. A function $\varphi : \overline{S} \to \overline{U}$ represents a *Markov or feedback control policy* from states in the augmented state space $\overline{S}$, which is known to be admissible with respect to the process noise $w(\cdot)$. Let Ψ be the set of all such policies φ. Let $\mu : \overline{S} \to U$ and $\kappa : \overline{S} \to \mathbb{R}^{d_w}$ so that $\varphi = (\mu, \kappa)$. We rename $T_{u,z}$ to $T_{\varphi,z}$ for the sake of notation clarity. Using these notations, $\mu(\cdot, 1)$ is thus a Markov control policy for the unconstrained problem, i.e. the problem without the risk constraint, that maps from S to U. Henceforth, we will *use* $\mu(\cdot)$ *to refer to* $\mu(\cdot, 1)$ when it is clear from the context. Let Π be the set of all such Markov control policies $\mu(\cdot)$ on S.

Now, let us rewrite the cost-to-go function $J_u(z)$ in Eq. (5.3) for the threshold η at time 0 in a new form:

$$J_\varphi(z, \eta) = \mathbb{E}\left[\int_0^{T_{\varphi,z}} \alpha^t g\big(x(t), \mu(x(t), q(t))\big)\, dt + \alpha^{T_{\varphi,z}} h(x(T_{\varphi,z})) \Big| (x, q)(0) = (z, \eta) \right]. \quad (5.7)$$

We therefore transform the risk-constrained problem in Eqs. (5.4)-(5.5) into a stochastic target problem as follows:

$$\mathcal{OPT}4: \quad J^*(z, \eta) = \inf_{\varphi \in \Psi} J_\varphi(z, \eta) \quad (5.8)$$

$$\text{s/t} \quad 1_\Gamma(x(T_{\varphi,z})) \leq q(T_{\varphi,z}) \;\; a.s. \;\; and \;\; Eq. \text{ (5.6)}. \quad (5.9)$$

We note that the comma in $J^*(z, \eta)$ signifies that η is now a state component rather than a parameter, and we can recognize that $J^*(z, \eta)$ is equal to $J^*(z; \eta)$ in $\mathcal{OPT}3$. The constraint in the above formulation specifies the relationship of random variables at the terminal time as a target set, and hence the name of this formulation [24,25][2]. In this formulation, we solve for feedback control policies φ for all $(z, \eta) \in \overline{S}$ instead of a particular choice of η for $x(0) = z$ at time $t = 0$.

We note that in this formulation, boundary conditions are not fully specified *a priori*. In the following subsection, we discuss how to remove the constraint in Eq. (5.9) by constructing its boundary and computing the boundary values.

[2] In [24,25], the authors use the name "stochastic target problems" to refer to feasibility problems without objective functions. With slight abuse of terminology, we use the same name for problems with objective functions.

5.2.3 Characterization and boundary conditions

The domain of $\mathcal{OPT}4$ is:

$$D = \{(z, \eta) \in \overline{S} \mid \exists \varphi \in \Psi \; s/t \; 1_\Gamma(x(T_{\varphi,z})) \leq q(T_{\varphi,z}) \; a.s.\}.$$

By the definition of the risk-constrained problem $\mathcal{OPT}3$, we can see that if $(z, \eta) \in D$ then $(z, \eta') \in D$ for any $\eta < \eta' \leq 1$. Thus, for each $z \in S$, we define

$$\gamma(z) = \inf \{\eta \in [0, 1] \mid (z, \eta) \in D\}, \tag{5.10}$$

as the infimum of risk tolerance at z. Therefore, we also have:

$$\gamma(z) = \inf_{u \in \mathcal{U}} P_0^z(x(T_{u,z}) \in \Gamma) = \inf_{u \in \mathcal{U}} \mathbb{E}\Big[1_\Gamma(x(T_{u,z})) \mid x(0) = z\Big]. \tag{5.11}$$

Thus, the boundary of D is

$$\partial D = S \times \{1\} \cup \{(z, \gamma(z)) \mid z \in S\} \cup \{(z, \eta) \mid z \in \partial S, \eta \in [\gamma(z), 1]\}. \tag{5.12}$$

For states in $\{(z, \eta) \mid z \in \partial S, \eta \in [\gamma(z), 1]\}$, the system stops on ∂S and takes terminal values according to $h(\cdot)$.

The domain D is illustrated in Fig. 5-1. In this example, the state space S is a bounded two-dimensional area with boundary ∂S containing a goal region G and an obstacle region $\Gamma = Obs$. The augmented state space $\overline{S}$ augments S with an extra dimension for the martingale state q. The infimum probability of reaching into Γ from states in S is depicted as γ. As we can see, γ takes value 1 in Γ. The volume between γ and the hyper-plane $q = 1$ is the domain D of $\mathcal{OPT}4$.

Now, let $\eta = 1$, we notice that $J^*(z, 1)$ is the optimal cost-to-go from z for the stochastic optimal problem without the risk constraint:

$$J^*(z, 1) = \inf_{u \in \mathcal{U}} J_u(z). \tag{5.13}$$

As seen in Chapter 3, an optimal control process that solves the optimization problem in Eq. (5.13) is given by a Markov policy $\mu^*(\cdot, 1) \in \Pi$. We now define the failure probability function $\Upsilon : S \to [0, 1]$ under such an optimal policy $\mu^*(\cdot, 1)$ as follows:

$$\Upsilon(z) = 1_\Gamma(x(T_{\mu^*,z})), \; \forall z \in S, \tag{5.14}$$

where $T_{\mu^*,z}$ is the first exit time when the system follows the control policy $\mu^*(\cdot, 1)$ from the initial state z. By the definitions of γ and Υ, we can recognize that $\Upsilon(z) \geq \gamma(z)$ for all $z \in S$. Figure 5-2 shows an illustration of Υ for the same example in Fig. 5-1.

Since following the policy $\mu^*(\cdot, 1)$ from an initial state z yields a failure probability $\Upsilon(z)$, we infer that:

$$J^*(z, 1) = J^*(z, \Upsilon(z)). \tag{5.15}$$

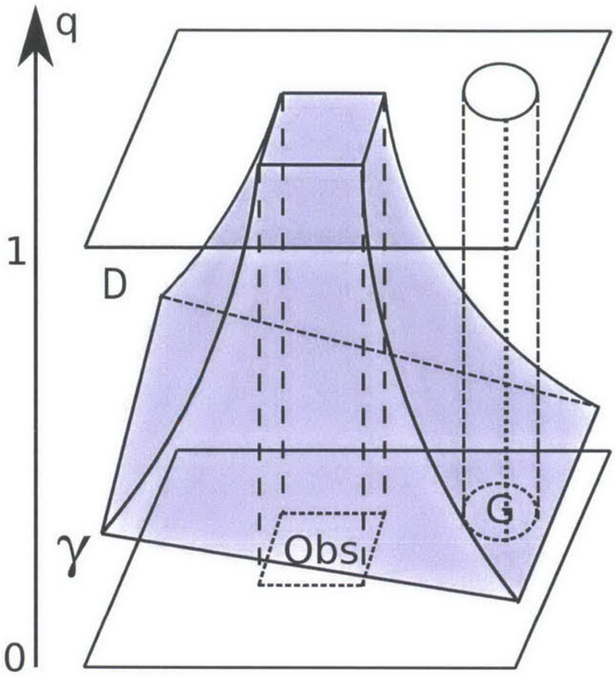


Figure 5-1: A domain of $\mathcal{OPT}4$. The state space S is a bounded two-dimensional area with boundary ∂S containing a goal region G and an obstacle region $\Gamma = Obs$. The augmented state space $\overline{S}$ augments S with an extra dimension for the martingale state q. The infimum probability of reaching into Γ from states in S is depicted as γ. γ takes value 1 in Γ. The volume between γ and the hyper-plane $q = 1$ is the domain D of $\mathcal{OPT}4$.

From the definition of the problem $\mathcal{OPT}3$, we also have:

$$0 \leq \eta < \eta' \leq 1 \Rightarrow J^*(z, \eta) \geq J^*(z, \eta'). \tag{5.16}$$

Thus, for any $\Upsilon(z) < \eta < 1$, we have:

$$J^*(z, 1) \leq J^*(z, \eta) \leq J^*(z, \Upsilon(z)). \tag{5.17}$$

Combining Eq. (5.15) and Eq. (5.17), we have:

$$\forall \ \eta \in [\Upsilon(z), 1] \Rightarrow J^*(z, \eta) = J^*(z, 1). \tag{5.18}$$

As a consequence, when we start from an initial state z with a risk threshold η that is at least $\Upsilon(z)$, it is optimal to execute an optimal control policy of the corresponding unconstrained problem from the initial state z.

It also follows from Eq. (5.16) that reducing the risk tolerance from 1.0 along the controlled process can not reduce the optimal cost-to-go function evaluated at $(x(t), q(t) = 1.0)$. Thus, we infer that for augmented states $(x(t), q(t))$ where $q(t) = 1.0$, the optimal martingale control $c^*(t)$ is 0.

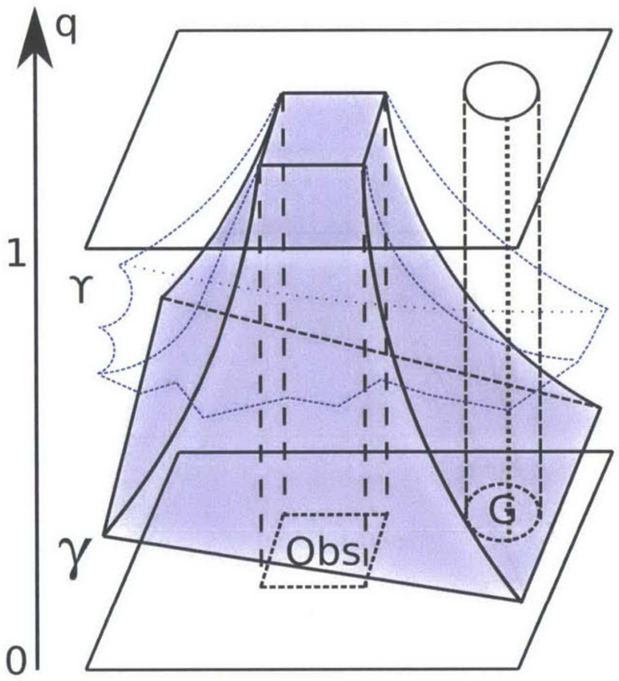


Figure 5-2: Illustration of the failure probability function Υ due to an optimal control policy $\mu^*(\cdot, 1)$ of the unconstrained problem. Continuing from the example in Fig. 5-1, we plot Υ for the same two-dimensional example. By the definitions of γ and Υ, we have $\Upsilon \geq \gamma$.

Now, under all admissible policies φ, we can not obtain a failure probability for an initial state z that are lower than $\gamma(z)$. Thus, it is clear that $J^*(z, \eta) = +\infty$ for all $0 \leq \eta < \gamma(z)$. The following lemma characterizes the optimal martingale control $c^*(t)$ for augmented states $(x(t), q(t) = \gamma(x(t)))$.

Lemma 5.2.2 *Given the problem definition as in Eqs.* (5.4)-(5.5). *We assume that* $\gamma(x)$ *is a smooth function*[3]. *When* $q(t) = \gamma(x(t))$ *and* $u(t)$ *is chosen, we must have:*

$$c(t)^T = \frac{\partial \gamma}{\partial x(t)}^T F(x(t), u(t)). \tag{5.19}$$

Proof Using the geometric dynamic programming principle (Theorem 2.2.13), we have the following result: starting from $q(t) = \gamma(x(t))$, for all stopping time $\tau \geq t$, a feasible control policy $\varphi \in \Psi$ satisfies

$$q(\tau) \geq \gamma(x(\tau))$$

almost surely.

[3] When $\gamma(x)$ is not smooth, we need the concept of viscosity solutions and weak dynamic programming principle. See [24, 25] for details.

Take $\tau = t+$, under a feasible control policy φ, we have $q(t+) \geq \gamma(x(t+))$ a.s. for all t, and hence $dq(t) \geq d\gamma(x(t))$ a.s. By Itô lemma (see Section 2.2.1), we derive the following relationship:

$$\begin{aligned} c^T(t)dw(t) \geq & \frac{\partial \gamma}{\partial x}^T \Big(f(x(t), u(t))dt + F(x(t), u(t))dw(t) \Big) \\ & + \frac{1}{2} Tr \Big(F(x(t), u(t)) F(x(t), u(t))^T \frac{\partial^2 \gamma}{(\partial x)^2} \Big) dt \ a.s. \end{aligned}$$

For the above inequality to hold almost surely, the coefficient of $dw(t)$ must be 0, i.e.:

$$c(t)^T - \frac{\partial \gamma}{\partial x(t)}^T F(x(t), u(t)) = 0.$$

This leads to Eq. (5.19). ■

In addition, if a control process that solves Eq. (5.11) is obtainable, say u_γ, the cost-to-go due to that control process is $J_{u_\gamma}(z)$. We will conveniently refer to $J_{u_\gamma}(z)$ as $J^\gamma(z)$. Under the mild assumption that u_γ is unique, it follows that $J^\gamma(z) = J^*(z, \gamma(z))$.

We also emphasize that when $(x(t), q(t))$ is inside the interior D^o of D, the usual dynamic programming principle holds. The extension of iMDP outlined below is designed to compute the sequence of approximate cost-to-go values on the boundary ∂D and in the interior D^o.

5.3 Algorithm

The following discussion follows closely the presentation in Chapters 3 and 4. Nevertheless, we will work with both the original state space S and the augmented state space $\overline{S}$. Thus, we will repeat the description in detail for the sake of clarity.

In particular, we briefly overview how the Markov chain approximation technique is used in both the original and augmented state spaces. We then present the extended iMDP algorithm that incrementally constructs the boundary values and computes solutions to our problem. We sample in the original state space S to compute $J^*(\cdot, 1)$ and its induced collision probability $\Upsilon(\cdot)$ as in Eq. (5.14), the min-failure probability $\gamma(\cdot)$ as in Eq. (5.11) and its induced cost-to-go $J^\gamma(\cdot)$. Concurrently, we also sample in the augmented state space $\overline{S}$ with appropriate values for samples on the boundary of D and approximate the optimal cost-to-go function $J^*(\cdot, \cdot)$ in the interior D^o. As a result, we construct a sequence of anytime control policies to approximate an optimal control policy $\varphi^* = (\mu^*, \kappa^*)$ in an efficient iterative procedure.

5.3.1 Markov chain approximation

On the state space S, we want to approximate $J^*(z, 1)$, $\Upsilon(z)$, $\gamma(z)$ and $J^\gamma(z)$ for any state $z \in S$, and it suffices to consider Markov controls as shown in [137, 138]. The Markov chain approximation method approximates the continuous dynamics in

Eq. (5.1) 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* as presented in Chapter 3. In particular, we construct $G_n(z, v) = g(z, v)\Delta t_n(z)$, and $H_n(z) = h(z)$ for each $z \in S_n$ and $v \in U$. We also require:

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

$$\begin{aligned}\lim_{n\to\infty} \frac{\mathbb{E}_{P_n}[\Delta \xi_i^n \mid \xi_i^n = z, u_i^n = v]}{\Delta t_n(z)} &= f(z, v),\\ \lim_{n\to\infty} \frac{\mathrm{Cov}_{P_n}[\Delta \xi_i^n \mid \xi_i^n = z, u_i^n = v]}{\Delta t_n(z)} &= F(z, v)F(z, v)^T.\end{aligned}$$

We recall that in the Markov chain approximation approach, we solve a sequence of control problems defined on $\{\mathcal{M}_n\}_{n=0}^{\infty}$ as follows. A Markov or feedback 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 that approximates a Markov control process $u(\cdot)$ in Eq. (5.3), the corresponding cost-to-go due to μ_n on $\mathcal{M}_n$ is:

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

where $\{\xi_i^n; i \in \mathbb{N}\}$ is the sequence of states of the controlled Markov chain under the policy μ_n, and I_n is the 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 *optimal cost-to-go function* $J_n^*(\cdot, 1) : S_n \to \overline{\mathbb{R}}$ that approximates the unconstrained optimal cost-to-go function $J^*(\cdot, 1)$ is denoted as

$$J_n^*(z, 1) = \inf_{\mu_n \in \Pi_n} J_{n,\mu_n}(z) \;\forall z \in S_n. \tag{5.20}$$

An *optimal policy* for the unconstrained problem in Eq. (5.20), denoted by μ_n^*, satisfies $J_{n,\mu_n^*}(z) = J_n^*(z, 1)$ for all $z \in S_n$. For any $\epsilon > 0$, μ_n is an ϵ-optimal policy if $||J_{n,\mu_n}(\cdot) - J_n^*(\cdot, 1)||_\infty \leq \epsilon$. We also define the failure probability function $\Upsilon_n : S_n \to [0, 1]$ due to an optimal policy μ_n^* as follows:

$$\Upsilon_n(z) = \mathbb{E}_{P_n}\left[1_\Gamma(\xi_{I_n}^n) \mid x(0) = z \;;\; \mu_n^* \;\right] \;\forall z \in S_n, \tag{5.21}$$

where we denote μ_n^* after the semicolon (as a parameter) to emphasize the dependence of the Markov chain on this control policy.

In addition, the *min-failure probability* γ_n on $\mathcal{M}_n$ that approximates γ is defined as:

$$\gamma_n(z) = \inf_{\mu_n \in \Pi_n} \mathbb{E}_{P_n}\left[1_\Gamma(\xi_{I_n}^n) \mid x(0) = z\right] \;\forall z \in S_n. \tag{5.22}$$

We note that the optimization programs in Eq. (5.20) and Eq. (5.22) may have two different optimal feedback control policies. Let $\nu_n \in \Pi_n$ be a control policy on $\mathcal{M}_n$ that achieves γ_n, then the cost-to-go function due to ν_n is J_{n,ν_n} which approximates J^{γ}. For this reason, we conveniently refer to J_{n,ν_n} as J_n^{γ}.

Similarly, in the augmented state space $\overline{S}$, we use a sequence of MDPs $\{\overline{\mathcal{M}}_n = (\overline{S}_n, \overline{U}, \overline{P}_n, \overline{G}_n, \overline{H}_n)\}_{n=0}^{\infty}$ and a sequence of holding times $\{\overline{\Delta t}_n\}_{n=0}^{\infty}$ that are locally consistent with the augmented dynamics in Eq. (5.6). In particular, $\overline{S}_n$ is a random subset of $D \subset \overline{S}$, $\overline{G}_n$ is identical to G_n, and $\overline{H}_n(z,\eta)$ is equal to $H_n(z)$ if $\eta \in [\gamma_n(z), 1]$ and $+\infty$ otherwise. Similar to the construction of P_n and Δt_n, we also construct the transition probabilities $\overline{P}_n$ on $\overline{\mathcal{M}}_n$ and holding time $\overline{\Delta t}_n$ that satisfy the local consistency conditions for nominal dynamics $\overline{f}(x, q, u, c)$ and dispersion matrix $\overline{F}(x, q, u, c)$.

A trajectory on $\overline{\mathcal{M}}_n$ is denoted as $\{\overline{\xi}_i^n; i \in \mathbb{N}\}$ where $\overline{\xi}_i^n \in \overline{S}_n$. A Markov policy φ_n is a function that maps each state $(z,\eta) \in \overline{S}_n$ to a control $(\mu_n(z,\eta), \kappa_n(z,\eta)) \in \overline{U}$. Moreover, admissible κ_n at $(z,1) \in \overline{S}_n$ is 0 and at $(z, \gamma_n(z)) \in \overline{S}_n$ is a function of $\mu(z, \gamma_n(z))$ as shown in Eq. (5.19). Admissible κ_n for other states in $\overline{S}_n$ is such that the martingale-component process of $\{\overline{\xi}_i^n; i \in \mathbb{N}\}$ belongs to [0,1] almost surely. Using the fact that Brownian motions can approximated as random walks, from Lemma 5.2.1, we can show that equivalently, each control component of $\kappa_n(z,\eta)$ belongs to

$$\left[-\frac{\min(\eta, 1-\eta)}{\overline{\Delta t}_n d_w}, \frac{\min(\eta, 1-\eta)}{\overline{\Delta t}_n d_w} \right]. \tag{5.23}$$

The set of all such policies φ_n is Ψ_n.

Under a control policy φ_n, the cost-to-go function on $\overline{\mathcal{M}}_n$ that approximates the function in Eq. (5.7) is defined as:

$$J_{n,\varphi_n}(z,\eta) = \mathbb{E}_{\overline{P}_n} \left[\sum_{i=0}^{\overline{I}_n - 1} \alpha^{\overline{t}_i^n} \overline{G}_n(\overline{\xi}_i^n, \mu_n(\overline{\xi}_i^n)) + \alpha^{\overline{t}_{\overline{I}_n}^n} \overline{H}_n(\overline{\xi}_{\overline{I}_n}^n) \;\Big|\; \overline{\xi}_0^n = (z,\eta) \right],$$

where $\overline{t}_i^n = \sum_0^{i-1} \overline{\Delta t}_n(\overline{\xi}_i^n)$ for $i \geq 1$ with $\overline{t}_0^n = 0$, and $\overline{I}_n$ is index when the x-component of $\overline{\xi}_i^n$ first arrives at ∂S. The approximating optimal cost $J_n^* : \overline{S}_n \to \overline{\mathbb{R}}$ for J^* in Eq. (5.8) is:

$$J_n^*(z,\eta) = \inf_{\varphi_n \in \Psi_n} J_{n,\varphi_n}(z,\eta) \;\; \forall (z,\eta) \in \overline{S}_n. \tag{5.24}$$

To solve the above optimization, we compute approximate boundary values for states on the boundary of D using the sequence of MDP $\{\mathcal{M}_n\}_{n=0}^{\infty}$ on S as discussed above. For states $(z,\eta) \in \overline{S}_n \cap D^o$, the normal dynamic programming principle holds.

The extension of iMDP outlined below is designed to compute the sequence of optimal cost-to-go functions $\{J_n^*\}_{n=0}^{\infty}$, associated failure probability functions $\{\Upsilon_n\}_{n=0}^{\infty}$, min-failure probability functions $\{\gamma_n\}_{n=0}^{\infty}$, min-failure cost functions $\{J_n^{\gamma}\}_{n=0}^{\infty}$, and the sequence of anytime control policies $\{\mu_n\}_{n=0}^{\infty}$ and $\{\kappa_n\}_{n=0}^{\infty}$ in an incremental procedure.

5.3.2 Extended iMDP algorithm

Before presenting the details of the algorithm, we discuss a number of primitive procedures.

Sampling

The $\texttt{Sample(X)}$ procedure sample states independently and uniformly in X.

Nearest Neighbors

Given $\zeta \in X \subset \mathbb{R}^{d_X}$ and a set $Y \subseteq X$, for any $k \in \mathbb{N}$, the procedure $\texttt{Nearest}(\zeta, Y, k)$ returns the k nearest states $\zeta' \in Y$ that are closest to ζ in terms of the d_X-dimensional Euclidean norm.

Time Intervals

Given a state $\zeta \in X$ and a number $k \in \mathbb{N}$, the procedure $\texttt{ComputeHoldingTime}(\zeta, k, d)$ returns a holding time computed as follows:

$$\texttt{ComputeHoldingTime}(\zeta, k, d) = \chi_t \left(\frac{\log k}{k} \right)^{\theta\varsigma\rho/d},$$

where $\chi_t > 0$ is a constant, and ς, θ are constants in $(0, 1)$ and $(0, 1]$ respectively.

Transition Probabilities

We are given a state $\zeta \in X$, a subset $Y \in X$, a control v in some control set V, a positive number τ describing a holding time, k is a nominal dynamics, K is a dispersion matrix. The procedure $\texttt{ComputeTranProb}(\zeta, v, \tau, Y, k, K)$ returns:

i. A finite set $Z_{\text{near}} \subset X$ of states such that the state $\zeta + k(\zeta, v)\tau$ belongs to the convex hull of Z_{near} and $||z' - z||_2 = O(\tau)$ for all $\zeta' \neq \zeta \in Z_{\text{near}}$, 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}.

As done in the previous chapters, these transition probabilities are designed to provide a sequence of locally consistent Markov chains that approximate the nominal dynamics k and the dispersion matrix K.

Backward Extension

Given $T > 0$ and two states $z, z' \in S$, the procedure $\texttt{ExtBackwardsS}(z, z', T)$ 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, the procedure returns failure. We can solve for the triple

Algorithm 7: Risk Constrained iMDP()

```
$(S_0, \overline{S}_0, J_0, \gamma_0, \Upsilon_0, J_0^\gamma, \mu_0, \kappa_0, \Delta t_0, \overline{\Delta t}_0) \leftarrow (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$;
for $n = 1 \rightarrow N$ do
    UpdateDataStorage($n-1, n$) ;
    SampleOnBoundary($n$) ;
    // $K_{1,n} \geq 1$ rounds to construct boundary conditions
    for $i = 1 \rightarrow K_{1,n}$ do
        ConstructBoundary($S_n, \overline{S}_n, J_n, \gamma_n, \Upsilon_n, J_n^\gamma, \mu_n, \Delta t_n$) ;
    // $K_{2,n} \geq 0$ rounds to process the interior region
    for $i = 1 \rightarrow K_{2,n}$ do
        ProcessInterior($S_n, \overline{S}_n, J_n, \gamma_n, \Upsilon_n, J_n^\gamma, \mu_n, \kappa_n, \overline{\Delta t}_n$);
```

(x, v, τ) by sampling several controls v and choose the control resulting in $x(0)$ that is closest to z'.

When $(z, \eta), (z', \eta')$ are in $\overline{S}$, the procedure ExtBackwardsSM($(z, \eta), (z', \eta'), T$) returns (x, q, v, τ) in which (x, v, τ) is the output of ExtBackwardsS(z, z', T) and q is sampled according to a Gaussian distribution $N(\eta', \sigma_q)$ where σ_q is a parameter.

Sampling and Discovering Controls

For $z \in S$ and $Y \subseteq S$, the procedure ConstructControlsS(k, z, Y, T) returns a set of k controls in U. We can uniformly sample k controls in U. Alternatively, for each state $z' \in$ 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 $x(T) = z'$.

For $(z, \eta) \in \overline{S}$ and $Y \subseteq \overline{S}$, the procedure ConstructControlsSM($k, (z, \eta), Y, T$) returns a set of k controls in $\overline{U}$ such that the U-component of these controls are computed as in ConstructControlsS, and the martingale-control-components of these controls are sampled in admissible sets.

Algorithm Description

The extended iMDP algorithm is presented in Algorithms 7-11. The algorithm incrementally refines two MDP sequences, namely $\{\mathcal{M}_n\}_{n=0}^\infty$ and $\{\overline{\mathcal{M}}_n\}_{n=0}^\infty$, and two holding time sequences, namely $\{\Delta t_n\}_{n=0}^\infty$ and $\{\overline{\Delta t}_n\}_{n=0}^\infty$, that consistently approximate the original system in Eq. (5.1) and the augmented system in Eq. (5.6) respectively. We associate with $z \in S_n$ a cost value $J_n(z, 1)$, a control $\mu_n(z, 1)$, a failure probability $\Upsilon_n(z)$ due to $\mu_n(\cdot, 1)$, a min-failure probability $\gamma_n(z)$, a cost-to-go value $J_n^\gamma(z)$ induced by the obtained min-failure policy. Similarly, we associate with $\overline{z} \in \overline{S}_n$ a cost value $J_n(\overline{z})$, a control $(\mu_n(\overline{z}), \kappa_n(\overline{z}))$.

As shown in Algorithm 7, initially, empty MDP models $\mathcal{M}_0$ and $\overline{\mathcal{M}}_0$ are created. The algorithm then executes N iterations in which it samples states on the pre-specified part of the boundary ∂D, constructs the un-specified part of ∂D and

Algorithm 8: ConstructBoundary($S_n, \overline{S}_n, J_n, \gamma_n, \Upsilon_n, J_n^\gamma, \mu_n, \Delta t_n$)

```
1  $z_s \leftarrow$ Sample($S$) ;
2  $z_{near} \leftarrow$ Nearest($z_s, S_n, 1$) ;
3  if $(x_e, u_e, \tau) \leftarrow$ ExtBackwardsS($z_{near}, z_s, T_0$) then
4      $z_e \leftarrow x_e(0)$;
5      $ic = \tau g(z_e, u_e) + \alpha^\tau J_n(z_{near}, 1)$;
6      $ic^\gamma = \tau g(z_e, u_e) + \alpha^\tau J_n^\gamma(z_{near})$;
7      $(S_n, \overline{S}_n, J_n(z_e, 1), \gamma_n(z_e), \Upsilon_n(z_e), J_n^\gamma(z_e), \mu_n(z_e, 1), \Delta t_n(z_e)) \leftarrow$
       $(S_n \cup \{z_e\}, \overline{S}_n \cup \{(z_e, 1)\}, ic, \gamma_n(z_{near}), \Upsilon_n(z_{near}), ic^\gamma, u_e, \tau)$ ;
       // Perform $L_n \geq 1$ updates
8      for $i = 1 \to L_n$ do
           // Choose $\mathcal{K}_n = \Theta(|S_n|^\theta) < |S_n|$ states
9          $Z_{update} \leftarrow$ Nearest($z_e, S_n \backslash \partial S_n, \mathcal{K}_n$) $\cup \{z_e\}$;
10         for $z \in Z_{update}$ do
11             UpdateS($z, S_n, J_n, \gamma_n, \Upsilon_n, J_n^\gamma, \mu_n, \Delta t_n$) ;
```

processes the interior of D. More specifically, at Line 3, UpdateDataStorage($n-1, n$) indicates that refined models in the n^{th} iteration are constructed from models in the $(n-1)^{th}$ iteration, which can be implemented by simply sharing memory among iterations. Using rejection sampling, the procedure SampleOnBoundary at Line 4 sample states in ∂S and $\partial S \times [0, 1]$ to add to S_n and $\overline{S}_n$ respectively. We also initialize appropriate cost values for these sampled states.

We conduct $K_{1,n}$ rounds to refine the MDP sequence $\{\mathcal{M}_n\}_{n=0}^\infty$ as done in the original iMDP algorithm using the procedure ConstructBoundary (Line 6). Thus, we can compute the cost function J_n and the associated failure probability function Υ_n on $S_n \times \{1\}$. In the same procedure, we compute the min-failure probability function γ_n as well as the min-failure cost function J_n^γ on S_n. In other words, the algorithm effectively constructs approximate boundaries for D and approximate cost-to-go functions J_n on these approximate boundaries over iterations. To compute cost values for the interior D^o of D, we conduct $K_{2,n}$ rounds of the procedure ProcessInterior (Line 8) that similarly refines the MDP sequence $\{\overline{\mathcal{M}}_n\}_{n=0}^\infty$ in the augmented state space. We can choose the values of $K_{1,n}$ and $K_{2,n}$ so that we perform a large number of iterations to obtain stable boundary values before processing the interior domain when n is small. In the following discussion, we will present in detail the implementations of these procedures.

In Algorithm 8, we show the implementation of the procedure ConstructBoundary. We construct a finer MDP model $\mathcal{M}_n$ based on the previous model as follows. A state z_s, is sampled from the interior of the state space S (Line 1). The nearest state z_{near} to z_s (Line 2) in the previous model is used to construct an extended state z_e by using the procedure ExtendBackwardsS at Line 3. The extended states z_e and $(z_e, 1)$ are added into S_n and $\overline{S}_n$ respectively. The associated cost value $J_n(z_e, 1)$, failure probability $\Upsilon_n(z_e)$, min-failure probability $\gamma_n(z_e)$, min-failure cost value $J_n^\gamma(z_e)$ and

Algorithm 9: ProcessInterior($S_n, \overline{S}_n, J_n, \gamma_n, \Upsilon_n, J_n^\gamma, \mu_n, \kappa_n, \overline{\Delta t}_n$)

```
1  $\overline{z}_s = (z_s, q_s) \leftarrow$ Sample($\overline{S}$);
2  $\overline{z}_{near} = (z_{near}, q_{near}) \leftarrow$ Nearest($\overline{z}_s, \overline{S}_n, 1$);
3  if $(x_e, q_e, u_e, \tau) \leftarrow$ ExtBackwardsSM($\overline{z}_{near}, \overline{z}_s, T_0$) then
4      $\overline{z}_e \leftarrow (x_e(0), q_e)$;
5      if $q_e < \gamma_n(z_{near})$ then
           // $\mathcal{C}$ takes a large value
6          $(\overline{S}_n, J_n(\overline{z}_e), \mu_n(\overline{z}_e), \kappa_n(\overline{z}_e), \overline{\Delta t}_n(\overline{z}_e)) \leftarrow (\overline{S}_n \cup \{\overline{z}_e\}, \mathcal{C}, u_e, 0, \tau)$ ;
7      else
8          $ic = \tau g(z_e, u_e) + \alpha^\tau J_n(\overline{z}_{near})$;
9          $(\overline{S}_n, J_n(\overline{z}_e), \mu_n(\overline{z}_e), \kappa_n(\overline{z}_e), \overline{\Delta t}_n(\overline{z}_e)) \leftarrow (\overline{S}_n \cup \{\overline{z}_e\}, ic, u_e, 0, \tau)$ ;
           // Perform $\overline{L}_n \geq 1$ updates
10         for $i = 1 \to \overline{L}_n$ do
               // Choose $\overline{K}_n = \Theta(|\overline{S}_n|^\theta) < |\overline{S}_n|$ states
11             $\overline{Z}_{\text{update}} \leftarrow$ Nearest($\overline{z}_e, \overline{S}_n \backslash \partial\overline{S}_n, \overline{K}_n$) $\cup \{\overline{z}_e\}$;
12             for $\overline{z} = (z, q) \in \overline{Z}_{\text{update}}$ do
13                 UpdateSM($\overline{z}, \overline{S}_n, J_n, \gamma_n, \Upsilon_n, J_n^\gamma, \mu_n, \kappa_n, \overline{\Delta t}_n$);
```

control $\mu_n(z_e)$ are initialized at Line 7.

We then perform $L_n \geq 1$ updating rounds in each iteration (Lines 8-11). In particular, we construct the update-set Z_{update} consisting of $K_n = \Theta(|S_n|^\theta)$ states and z_e where $|K_n| < |S_n|$. For each state z in Z_{update}, the procedure `UpdateS` as shown in Algorithm 10 implements the following Bellman update:

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

The details of the implementation are as follows. A set of U_n controls is constructed using the procedure `ConstructControlsS` 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 $P_n(\cdot \mid z, v)$ consistently over Z_{near} from the procedure `ComputeTranProb` (Line 4). The cost values for the state z and controls in U_n are computed at Lines 5. We finally choose the best control in U_n that yields the smallest updated cost value (Line 7). Correspondingly, we improve the min-failure probability γ_n and its induced min-failure cost value J_n^γ in Lines 9-12.

Similarly, in Algorithm 9, we carry out the sampling and extending process in the augmented state space $\overline{S}$ to refine the MDP sequence $\overline{\mathcal{M}}_n$ (Lines 1-3). In this procedure, if an extended node has a martingale state that is below the corresponding min-failure probability, we initialize the cost value for extended node with a very large constant $\mathcal{C}$ representing $+\infty$ (see Lines 5-6). Otherwise, we initialize the extended node as seen in Lines 8-9. We then execute $\overline{L}_n$ rounds (Lines 10-13) to update the

```
Algorithm 10: UpdateS($z, S_n, J_n, \gamma_n, \Upsilon_n, J_n^\gamma, \mu_n, \Delta t_n$)
$\tau \leftarrow$ ComputeHoldingTime($z, |S_n|, d_x$);
// Sample or discover $M_n = \Theta(\log(|S_n|))$ controls
$U_n \leftarrow$ ConstructControlsS($M_n, z, S_n, \tau$);
for $v \in U_n$ do
    $(Z_{\text{near}}, P_n) \leftarrow$ ComputeTranProb($z, v, \tau, S_n, f, F$);
    // Update cost
    $J \leftarrow \tau g(z, v) + \alpha^\tau \sum_{y \in Z_{\text{near}}} P_n(y) J_n(y, 1)$;
    if $J < J_n(z, 1)$ then
        $p \leftarrow \sum_{y \in Z_{\text{near}}} P_n(y) \Upsilon_n(y)$;
        $(J_n(z, 1), \Upsilon_n(z), \mu_n(z, 1), \Delta t_n(z)) \leftarrow (J, p, v, \tau)$;
    // Update min-failure probability
    $b \leftarrow \sum_{y \in Z_{\text{near}}} P_n(y) \gamma_n(y)$;
    if $b < \gamma_n(z)$ then
        $J \leftarrow \tau g(z, v) + \alpha^\tau \sum_{y \in Z_{\text{near}}} P_n(y) J_n^\gamma(y)$;
        $(\gamma_n(z), J_n^\gamma(z)) \leftarrow (b, J)$;
```

cost-to-go J_n for *states in the interior* D^o *of* D using the procedure UpdateSM as shown in Algorithm 11. When a state $\overline{z} \in \overline{S}_n$ is updated in UpdateSM, we perform the following Bellman update:

$$J_n(\overline{z}) = \min_{(v,c) \in \overline{U}_n(z)} \{\overline{G}_n(z, v) + \alpha^{\overline{\Delta t}_n(z)} \mathbb{E}_{\overline{P}_n}[J_{n-1}(\overline{y}) | \overline{z}, (v, c)]\},$$

where the control set $\overline{U}_n$ is constructed by the procedure ConstructControlsSM, and the transition probability $\overline{P}_n(\cdot | \overline{z}, (v, c))$ consistently approximates the augmented dynamics in Eq. (5.6). To implement the above Bellman update at Line 5 in Algorithm 11, we make use of the characteristics presented in Section 5.2.3 where the notation 1_A is 1 if the event A occurs and 0 otherwise. That is, when the martingale state s of a state $\overline{y} = (y, s)$ in the support $\overline{Z}_{near}$ is at least $\Upsilon_n(y)$, we substitute $J_n(\overline{y})$ with $J_n(y, 1)$. Similarly, when the martingale state s is equal to $\gamma_n(y)$, we substitute $J_n(\overline{y})$ with $J_n^\gamma(y)$.

Feedback control

At the n^{th} iteration, given a state $x \in S$ and a martingale component q, to find a policy control (v, c), we perform a Bellman update based on the approximated cost-to-go J_n for the augmented state (x, q). During the holding time $\overline{\Delta t}_n$, the original system takes the control v and evolves in the original state space S while we simulate the dynamics of the martingale component under the martingale control c. After this holding time period, the augmented system has a new state (x', q'), and we repeat the above process.

```
Algorithm 11: UpdateSM($\overline{z} = (z, q), \overline{S}_n, J_n, \gamma_n, \Upsilon_n, J_n^{\gamma}, \mu_n, \kappa_n, \overline{\Delta t}_n$)
$\overline{\tau} \leftarrow$ ComputeHoldingTime($\overline{z}, |\overline{S}_n|, d_x + 1$);
// Sample or discover $\overline{M}_n = \Theta(\log(|\overline{S}_n|))$ controls
$\overline{U}_n \leftarrow$ ConstructControlsSM($\overline{M}_n, \overline{z}, \overline{S}_n, \overline{\tau}$);
for $\overline{v} = (v, c) \in \overline{U}_n$ do
    $(\overline{Z}_{\text{near}}, \overline{P}_n) \leftarrow$ ComputeTranProb($\overline{z}, \overline{v}, \overline{\tau}, \overline{S}_n, \overline{f}, \overline{F}$);
    $J \leftarrow \tau g(z, v) + \alpha^{\tau} \sum_{\overline{y}=(y,s)\in\overline{Z}_{\text{near}}} \overline{P}_n(\overline{y}) \big[ 1_{s=\gamma_n(y)} J_n^{\gamma}(y) + 1_{\gamma_n(y)<s<\Upsilon_n(y)} J_n(\overline{y}) + 1_{s\geq\Upsilon_n(y)} J_n(y, 1) \big]$;
    // Improved cost
    if $J < J_n(\overline{z})$ then
        $(J_n(\overline{z}), \mu_n(\overline{z}), \kappa_n(\overline{z}), \overline{\Delta t}_n(\overline{z})) \leftarrow (J, v, c, \tau)$;
```

Figure 5-3 visualizes how feedback policies look in the original and augmented state spaces. In the augmented state space $\overline{S}$, a feedback control policy is a deterministic Markov policy as a function of an augmented state (x, q). As the system actually evolves in the original state space S, and the martingale state q can be seen as a random parameter at each state x, the feedback control policy is a randomized policy.

Using the characteristics presented in Section 5.2.3, we infer that when a certain condition meets, the system can start following a deterministic control policy. More precisely, we recall that for all $\eta \in [\Upsilon(z), 1]$, we have $J^*(z, \eta) = J^*(z, 1)$. Thus, starting from any augmented state (z, η) where $\eta > \Upsilon(z)$, we can solve the problem as if the failure probability were 1.0 and use optimal control policies of the unconstrained problem from the state z. We illustrate this idea in Fig. 5-4. As we can see, when the martingale state along the trajectory is at least the corresponding value provided by Υ, the system starts following a deterministic control policy $\mu_n(\cdot, 1)$ of the unconstrained problem.

Algorithm 12 implements the above feedback policy. As shown in this algorithm, Line 3 returns a deterministic policy of the unconstrained problem if the martingale state is large enough, and Lines 5-13 perform a Bellman update to find the best augmented control if otherwise. When the system starts using deterministic policies of the unconstrained problem, we can set the martingale state to 1.0 and set the optimal martingale control to 0 in the following control period.

Complexity

Similar to the original iMDP version in Chapter 3, the time complexity per iteration of the implementation in Algorithms 7-11 is $O\big(|\overline{S}_n|^{\theta}(\log|\overline{S}_n|)^2\big)$. The space complexity of the iMDP algorithm is $O(|\overline{S}_n|)$ where $|\overline{S}_n| = \Theta(n)$ due to our sampling strategy.

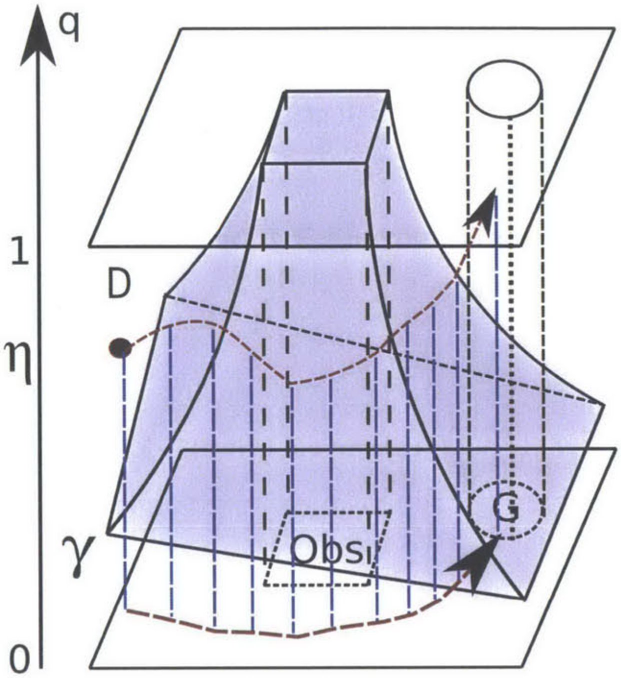


Figure 5-3: A feedback-controlled trajectory of $\mathcal{OPT}3$. In the augmented state space $\overline{S}$, a feedback control policy is a deterministic Markov policy as a function of an augmented state (x, q). As the system actually evolves in the original state space S, and the martingale state q can be seen as a random parameter at each state x, the feedback control policy is a randomized policy.

5.4 Analysis

Previous results in Chapter 3 show that $J_n(\cdot, 1)$ returned from the iMDP algorithm converges uniformly to $J^*(\cdot, 1)$ in probability. That is, we are able to compute $J^*(\cdot, 1)$ in an incremental manner without directly computing $J_n^*(\cdot, 1)$. As a consequence, it follows that Υ_n converges to Υ uniformly in probability. Using the same proof, we conclude that $\gamma_n(\cdot)$ and $J_n^\gamma(\cdot)$ converges uniformly to $\gamma(\cdot)$ and $J^*(\cdot, \gamma)$ in probability respectively. Therefore, we have incrementally constructed the boundary values on ∂D of the equivalent stochastic target problem presented in Eqs. (5.8)-(5.9). These results are established based on the approximation of the dynamics in Eq. (3.1) using the MDP sequence $\{\mathcal{M}_n\}_{n=0}^\infty$.

Similarly, the uniform convergence of $J_n(\cdot, \cdot)$ to $J^*(\cdot, \cdot)$ in probability on the interior of D is followed from the approximation of the dynamics in Eq. (5.6) using the MDP sequence $\{\overline{\mathcal{M}}_n\}_{n=0}^\infty$. In the following theorem, we formally summarize the key convergence results of the extended iMDP algorithm.

Theorem 5.4.1 *Let $\mathcal{M}_n$ and $\overline{\mathcal{M}}_n$ be two MDPs with discrete states constructed in S and $\overline{S}$ respectively, and let $J_n : \overline{S}_n \to \overline{\mathbb{R}}$ be the cost-to-go function returned by the extended iMDP algorithm at the n^{th} iteration. Let us define $||b||_X = \sup_{z \in X} b(z)$ as*

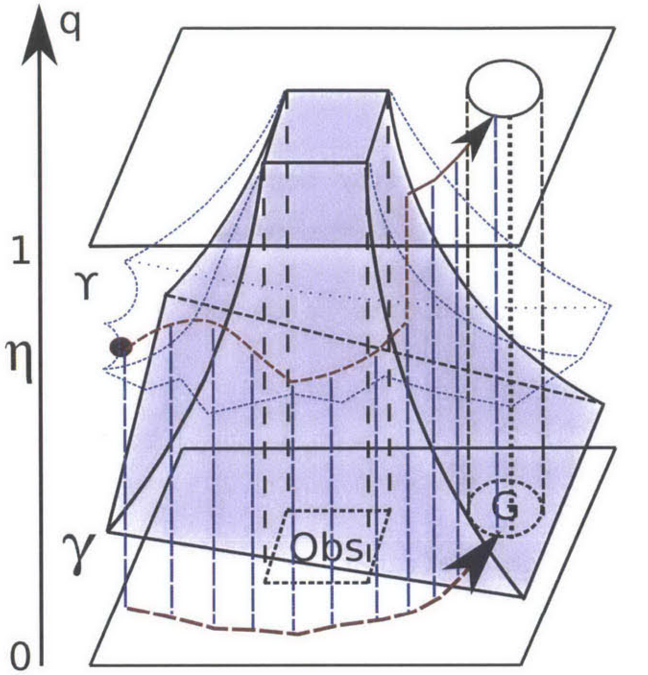


Figure 5-4: A modified feedback-controlled trajectory of $\mathcal{OPT}3$. We continue the illustration in Fig. 5-3. When the martingale state along the trajectory is at least the corresponding value provided by Υ, the system starts following a deterministic control policy $\mu_n(\cdot, 1)$ of the unconstrained problem.

the sup-norm over a set X of a function b with a domain containing X. We have the following random variables converge in probability:

1. $\text{plim}_{n\to\infty}||J_n(\cdot,1) - J^*(\cdot,1)||_{S_n} = 0,$

2. $\text{plim}_{n\to\infty}||\Upsilon_n - \Upsilon||_{S_n} = 0,$

3. $\text{plim}_{n\to\infty}||\gamma_n - \gamma||_{S_n} = 0,$

4. $\text{plim}_{n\to\infty}||J_n^\gamma - J^\gamma||_{S_n} = 0,$

5. $\text{plim}_{n\to\infty}||J_n - J^*||_{\overline{S}_n} = 0.$

The first four events construct the boundary values on ∂D in probability, which leads to the probabilistically sound property of the extended iMDP algorithm. The last event asserts the asymptotically optimal property through the convergence of the approximating cost-to-go function J_n to the optimal cost-to-go function J^* on the augmented state space $\overline{S}$.

Algorithm 12: Risk Constrained Policy($\overline{z} = (z, q) \in \overline{S}, n$)

```
$z_{\text{nearest}} \leftarrow$ Nearest($z, S_n, 1$);
if $q \geq \gamma_n(z_{\text{nearest}})$ then
    // Switch to a deterministic control policy
    return $\big(\varphi(\overline{z}) = (\mu_n(z_{\text{nearest}}), 0), \Delta t_n(z_{\text{nearest}})\big)$ ;
else
    // Perform a Bellman update to select a control
    $(J_{min}, v_{min}, c_{min}) \leftarrow (+\infty, \emptyset, \emptyset)$ ;
    $\overline{\tau} \leftarrow$ ComputeHoldingTime($\overline{z}$, $|\overline{S}_n|$, $d_x + 1$);
    // Sample or discover $\overline{M}_n = \Theta(\log(|\overline{S}_n|))$ controls
    $\overline{U}_n \leftarrow$ ConstructControlsSM($\overline{M}_n, \overline{z}, \overline{S}_n, \overline{\tau}$);
    for $\overline{v} = (v, c) \in \overline{U}_n$ do
        $(\overline{Z}_{\text{near}}, \overline{P}_n) \leftarrow$ ComputeTranProb($\overline{z}, \overline{v}, \overline{\tau}, \overline{S}_n, \overline{f}, \overline{F}$);
        $J \leftarrow \tau g(z, v) + \alpha^{\tau} \sum_{\overline{y}=(y,s) \in \overline{Z}_{\text{near}}} \overline{P}_n(\overline{y}) \big[1_{s=\gamma_n(y)} J_n^{\gamma}(y) +$
                $1_{\gamma_n(y)<s<\Upsilon_n(y)} J_n(\overline{y}) + 1_{s \geq \Upsilon_n(y)} J_n(y, 1)\big]$;
        // Improved cost
        if $J < J_{min}$ then
            $(J_{min}, v_{min}, c_{min}) \leftarrow (J, v, c)$ ;
    return $\big(\varphi(\overline{z}) = (v, c), \tau\big)$ ;
```

5.5 Experiments

We carried out an experiment that is similar to the experiment in Chapter 4. We controlled a system with stochastic single integrator dynamics to a goal region with free ending time in a cluttered environment (see Fig. 5-5). The dynamics is given by $dx(t) = u(t)dt + Fdw(t)$ where $x(t) \in \mathbb{R}^2$, $u(t) \in \mathbb{R}^2$, and $F = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$. The system stops when it collides with obstacles or reach the goal region. The cost function is the weighted sum of total energy spent to reach the goal G at $(8, 8)$, which is measured as the integral of square of control magnitude, and a terminal cost, which is -1000 for the goal region G and 10 for the obstacle region Γ, with a discount factor $\alpha = 0.9$. The maximum velocity of the system in the x and y directions is one. At the beginning, the system starts from $(6.5, -3)$. Failure is defined as collisions with obstacles, and thus we use *failure probability* and *collision probability* interchangeably.

We first show how the extended iMDP algorithm constructs the sequence of approximating MDPs on S over iterations in Fig. 5-6. In particular, Figs. 5-6(a)-5-6(c) depict anytime policies on the boundary $S \times 1.0$ after 500, 1000, and 3000 iterations. Figures 5-6(d)-5-6(f) show the Markov chains created by anytime policies found by the algorithm on $\mathcal{M}_n$ after 200, 500 and 1000 iterations. We observe that the structures of these Markov chains are indeed random graphs that are (asymptotically almost-surely) connected to cover the state space S. As in the original version of iMDP,

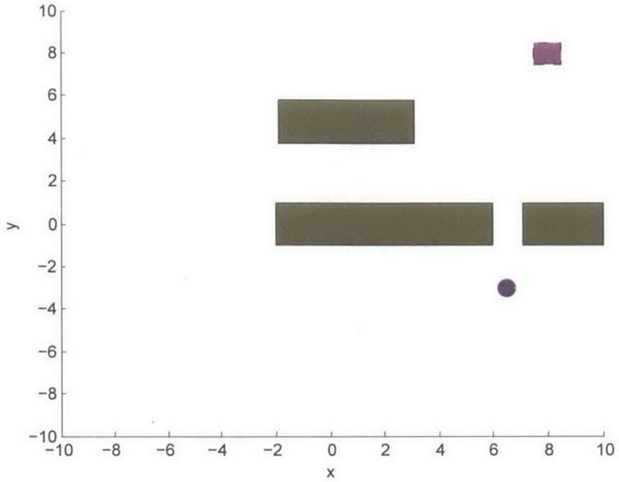


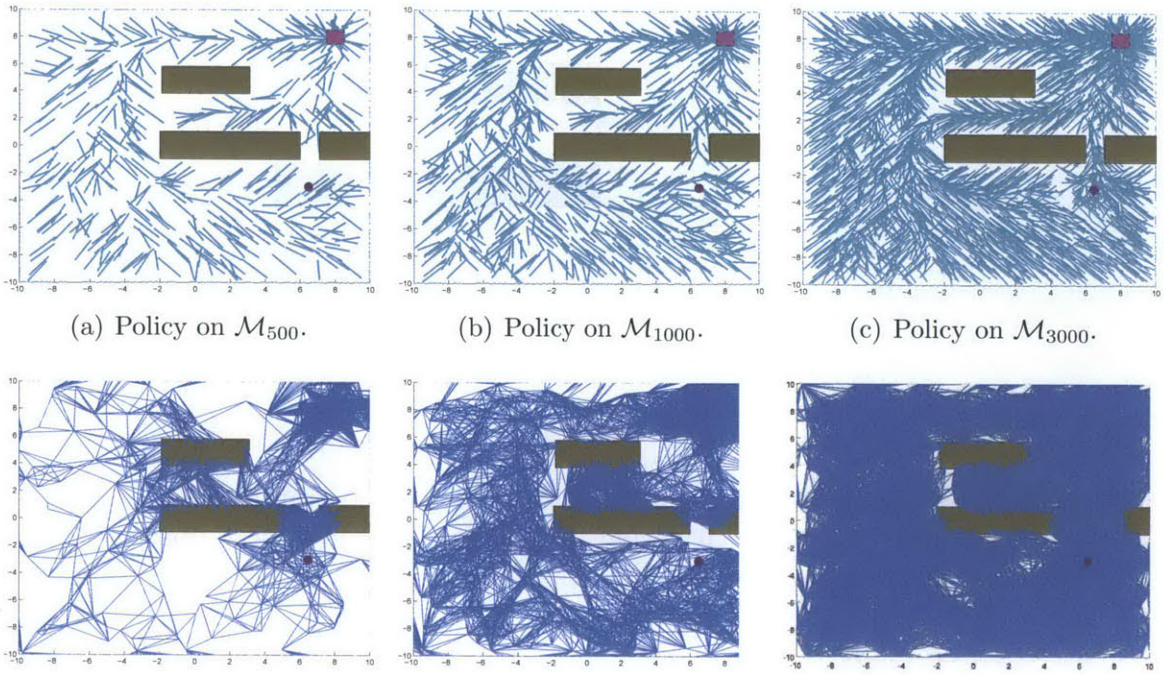
Figure 5-5: An operating environment for the single integrator system. The system starts at $(6.5, -3)$ to reach a goal at $(8, 8)$. There are three obstacles in the environment which creates narrow corridors.

it is worth noting that the structures of these Markov chains can be constructed on-demand during the execution of the algorithm.

The sequence of approximating MDPs on S provides boundary values for the stochastic target problem as shown in Fig. 5-7. In particular, Figs. 5-7(a)-5-7(c) shows a policy map, cost value function $J_{4000,1.0}$ and the associated collision probability function Υ_{4000} for the unconstrained problem after 4000 iterations. Similarly, Figs. 5-7(d)-5-7(f) show a policy map, the associated value function J^{γ}_{4000}, and the min-collision probability function γ_{4000} after 4000 iterations. As we can see, for the unconstrained problem, the policy map encourages the system to go through the narrow corridors with low cost-to-go values and high probabilities of collision. In contrast, the policy map from the min-collision probability problem encourages the system to detour around the obstacles with high cost-to-go values and low probabilities of collision.

We now show how the extended iMDP algorithm constructs the sequence of approximating MDPs on the augmented state space $\overline{S}$. Figures 5-8(a)-5-8(c) show the corresponding anytime policies in $\overline{S}$ over iterations. In Fig. 5-8(c), we show the top-down view of a policy for states in $\overline{\mathcal{M}}_{3000} \backslash \mathcal{M}_{3000}$. Compared to Fig 5-6(c), we observe that the system will try to avoid the narrow corridors when the risk tolerance is low. In Figs. 5-8(d)-5-8(f), we show the Markov chains that are created by anytime policies in the augmented state space. As we can see again, the structures of these Markov chains quickly cover $\overline{S}$ with (asymptotically almost-surely) connected random graphs.

We then examine how the algorithm computes the value functions for the interior D^o of the reformulated stochastic target problem in comparison with the value function of the unconstrained problem in Fig. 5-9. Figure 5-9(a)-5-9(c) show approximate cost-to-go J_n when the probability threshold η_0 is 1.0 for $n = 200$, 2000 and 4000. We recall that the value functions in these figures form the boundary conditions on $S \times 1$, which is a subset of ∂D. In the interior D^o, Figs. 5-9(d)-5-9(f) present the ap-



(a) Policy on $\mathcal{M}_{500}$.

(b) Policy on $\mathcal{M}_{1000}$.

(c) Policy on $\mathcal{M}_{3000}$.

(d) Markov chain implied by $\mathcal{M}_{200}$.

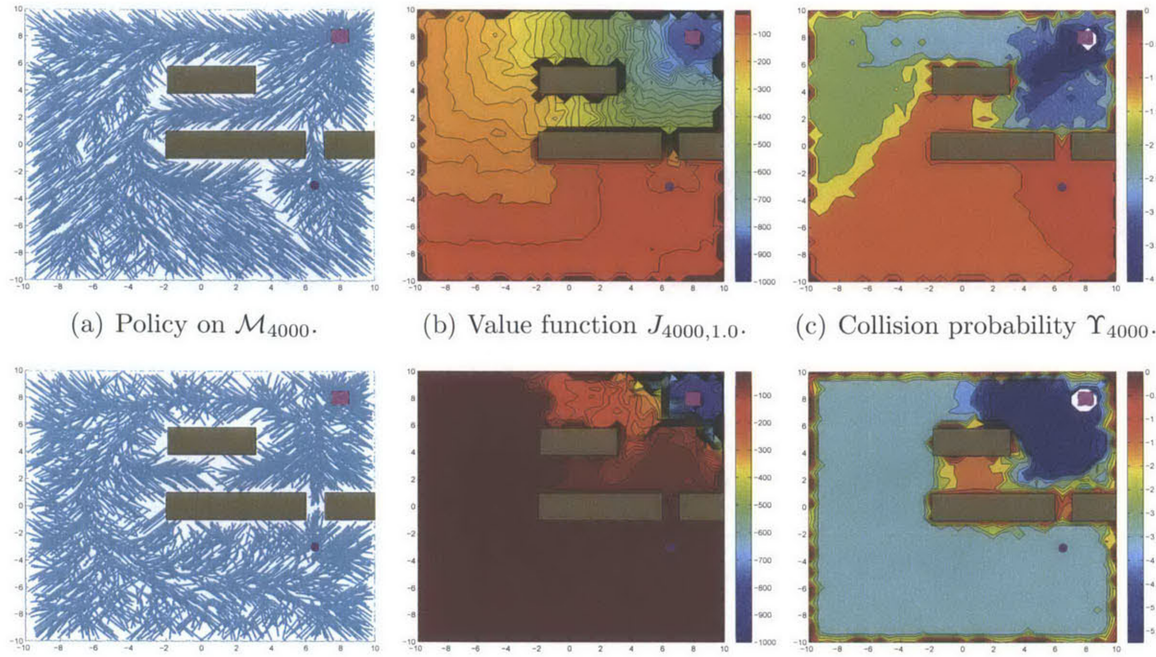
(e) Markov chain implied by $\mathcal{M}_{500}$.

(f) Markov chain implied by $\mathcal{M}_{1000}$.

Figure 5-6: A system with stochastic single integrator dynamics in a cluttered environment. The standard deviation of noise in x and y directions is 0.5. The cost function is the sum of total energy spent to reach the goal, which is measured as the integral of square of control magnitude, and a terminal cost, which is -1000 for the goal region (G) and 10 for the obstacle region (Γ), with a discount factor $\alpha = 0.9$. Figures 5-6(a)-5-6(c) depict anytime policies on the boundary $S \times 1.0$ over iterations. Figures 5-6(d)-5-6(f) show the Markov chains created by anytime policies on $\mathcal{M}_n$ over iterations.

proximate cost-to-go J_{4000} for augmented states where their martingale components are 0.1, 0.5 and 0.9. As we can see, the lower the martingale state is, the higher the cost value is – which is consistent with the characteristics in Section 5.2.3.

Lastly, we tested the performance of obtained anytime policies after 4000 iterations with different initial collision probability thresholds η. To do this, we first show how the policies of the unconstrained problem and the min-collision probability problem perform in Fig. 5-10. As we can see, in the unconstrained problem, the system takes risk to go through one of the narrow corridors to reach the goal. In contrast, in the min-collision probability problem, the system detour around the obstacles to reach the goal. While there are about 49.27% of 2000 trajectories (plotted in red) that collide with the obstacles for the former, we observe no collision out of 2000 trajectories for the latter. From the characteristics presented in Section 5.2.3 and illustrated in Fig. 5-4, from the starting state $(6.5, -3)$, for any initial collision probability threshold η that is at least 0.4927, we can execute the deterministic policy of the unconstrained



(a) Policy on $\mathcal{M}_{4000}$. (b) Value function $J_{4000,1.0}$. (c) Collision probability Υ_{4000}.

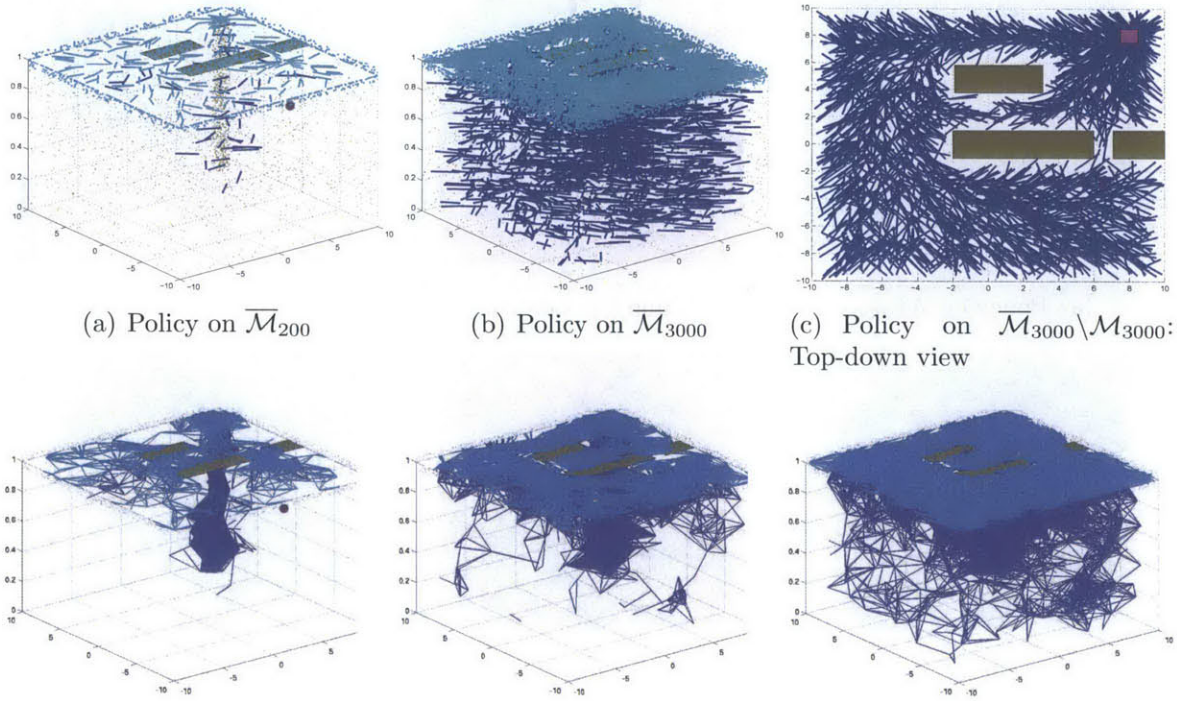
(d) Policy map induced by γ_{4000}. (e) Value function J^{γ}_{4000}. (f) Min-collision prob. γ_{4000}.

Figure 5-7: Boundary values. Figures 5-7(a)-5-7(c) shows a policy map, cost value function and the associated collision probability function for the unconstrained problem after 4000 iterations. Similar, Figures 5-7(d)-5-7(f) show a policy map, the associated value function, and the min-collision probability function after 4000 iterations. These values provide the boundary values for the stochastic target problem. For the unconstrained problem, the policy map encourages the system to go through the narrow corridors with low cost-to-go values and high probabilities of collision. In contrast, the policy map from the min-collision probability problem encourages the system to detour around the obstacles with high cost-to-go values and low probabilities of collision.

problem.

In Fig. 5-11, we provide an example of controlled trajectories that are illustrated in Fig. 5-4 when the system starts from $(6.5, -3)$ with the failure probability threshold $\eta = 0.4$. In this figure, the min-collision probability function γ_{4000} is plotted in blue, and the collision probability function Υ_{4000} is plotted in green. Starting from the augmented state $(6.5, -3, 0.40)$, the martingale state varies along controlled trajectories as a random parameter in a randomized control policy. When the martingale state is above Υ_{4000}, the system follows a deterministic control policy obtained from the unconstrained problem.

Similarly, in Fig. 5-12, we show controlled trajectories for different values of η $(0.01, 0.05, 0.10, 0.20, 0.30, 0.40)$. In Figs. 5-12(a)-5-12(c) and Figs. 5-12(g)-5-12(i), we show 50 trajectories resulting from a policy induced by J_{4000} with different initial collision probability thresholds. In Figs. 5-12(d)-5-12(f) and Figs. 5-12(j)-5-12(l),



(a) Policy on $\overline{\mathcal{M}}_{200}$

(b) Policy on $\overline{\mathcal{M}}_{3000}$

(c) Policy on $\overline{\mathcal{M}}_{3000} \backslash \mathcal{M}_{3000}$: Top-down view

(d) Markov chain implied by $\overline{\mathcal{M}}_{200}$.

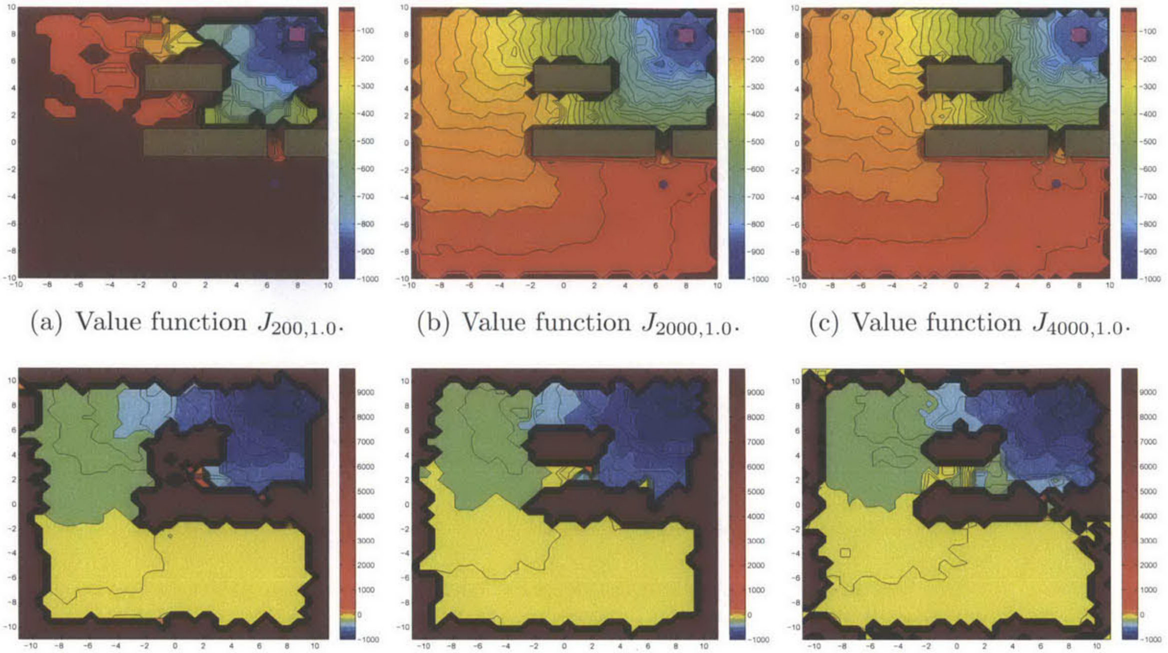
(e) Markov chain implied by $\overline{\mathcal{M}}_{500}$.

(f) Markov chain implied by $\overline{\mathcal{M}}_{1000}$.

Figure 5-8: Figures 5-8(a)-5-8(c) and Figures 5-8(d)-5-8(f) show the corresponding anytime policies and the associated Markov chains on $\overline{\mathcal{M}}_n$ respectively. In Fig. 5-8(c), we show the top-down view of a policy for states in $\overline{\mathcal{M}}_{3000} \backslash \mathcal{M}_{3000}$. We observe that the system will try to avoid the narrow corridors when the risk tolerance is low. We can also observe that the structures of the Markov chains quickly cover the state spaces S and $\overline{S}$ with connected random graphs.

we show 5000 corresponding trajectories in the original state space S with reported *simulated collision probabilities* and *average costs* in their captions. Trajectories that reach the goal region are plotted in blue, and trajectories that hit obstacles are plotted in red. These simulated collision probabilities and average costs are shown in Table 5.1. As we can see, the lower the threshold is, the higher the average cost is as we expect. When $\eta = 0.01$, the average cost is -19.42 and when $\eta = 1.0$, the average cost is -125.20.

More importantly, the simulated collision probabilities follow very closely the values of η chosen at time 0. In Fig. 5-13, we plot these simulated probabilities for the first N trajectories where $N \in [1, 5000]$ to show that the algorithm fully respects the bounded failure probability. Thus, this observation indicates that the extended iMDP algorithm is able to manage the risk tolerance along trajectories in different executions to minimize the expected costs using feasible and time-consistent anytime policies.



(a) Value function $J_{200,1.0}$.

(b) Value function $J_{2000,1.0}$.

(c) Value function $J_{4000,1.0}$.

(d) Value function $J_{4000,0.1}$

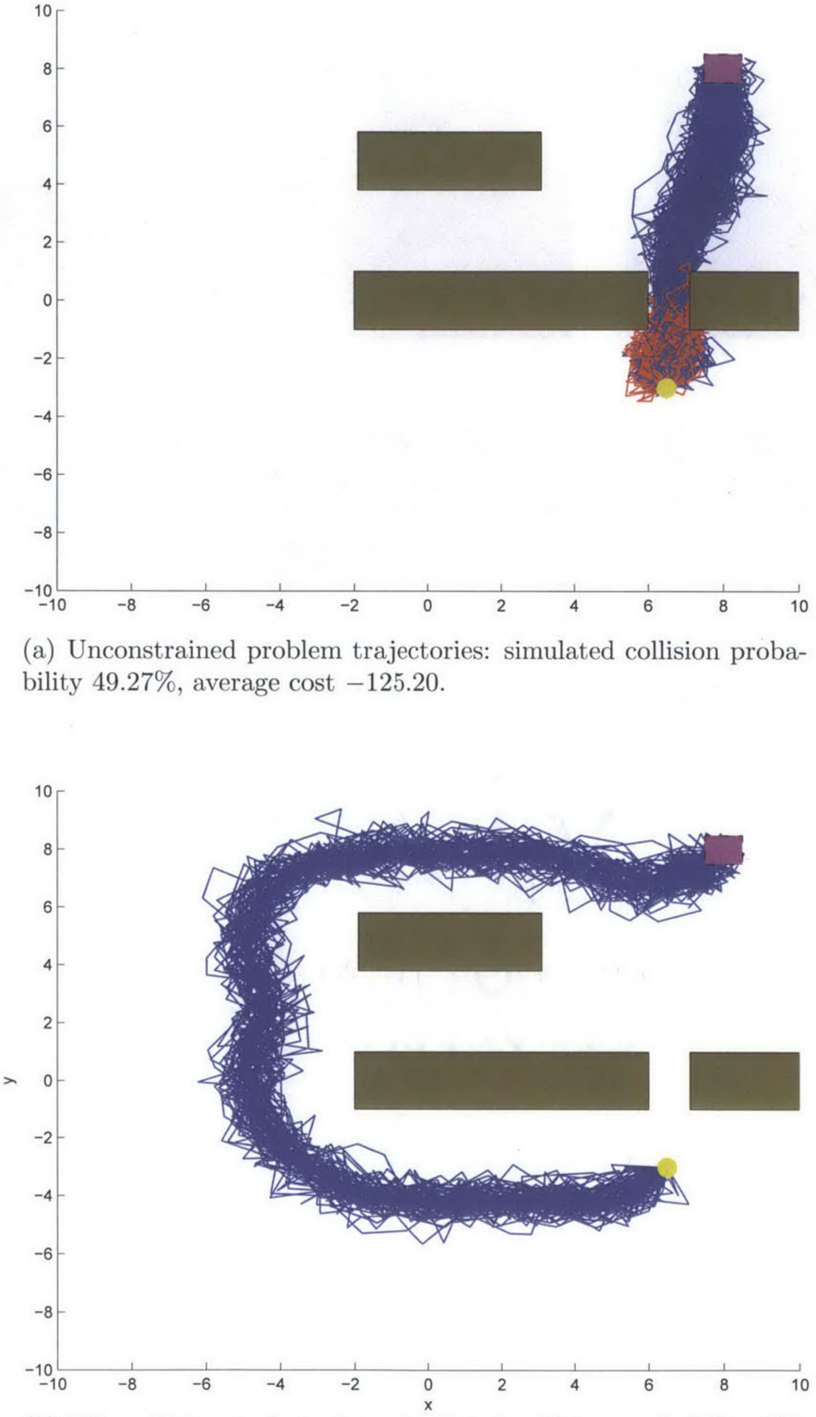
(e) Value function $J_{4000,0.5}$

(f) Value function $J_{4000,0.9}$

Figure 5-9: Examples of incremental value functions over iterations. Figure 5-9(a)-5-9(c) show the approximate cost-to-go functions J_n when the probability threshold η_0 is 1.0 for n = 200, 2000 and 4000. Figures 5-9(d)-5-9(f) present the approximate cost-to-go function J_{4000} in $\overline{\mathcal{M}}_{4000}$ for augmented states where their martingale components are 0.1, 0.5 and 0.9 respectively. The plot shows that the lower the martingale state is, the higher the cost value is – which is consistent with the characteristics in Section 5.2.3.

Table 5.1: Failure ratios and average costs for 5000 trajectories for Fig. 5-13.

η	**Failure Ratio**	**Average Cost**
1.00	0.4927	-125.20
0.40	0.4014	-115.49
0.30	0.2819	-76.80
0.20	0.1560	-65.81
0.10	0.1024	-58.00
0.05	0.0420	-42.53
0.01	0.0084	-19.42
0.001	0.0000	-18.86



(a) Unconstrained problem trajectories: simulated collision probability 49.27%, average cost -125.20.

(b) Min-collision trajectories: simulated collision probability 0%, average cost -17.85.

Figure 5-10: Examples of trajectories from policies of the unconstrained problem (Fig. 5-10(a)) and the min-collision probability problem (Fig. 5-10(b)). In the unconstrained problem, the system takes risk to go through one of the narrow corridors to reach the goal. In contrast, in the min-collision probability problem, the system detours around the obstacles to reach the goal. While there are about 49.27% of 2000 trajectories (plotted in red) that collide with the obstacles for the former, we observe no collision out of 2000 trajectories for the latter.

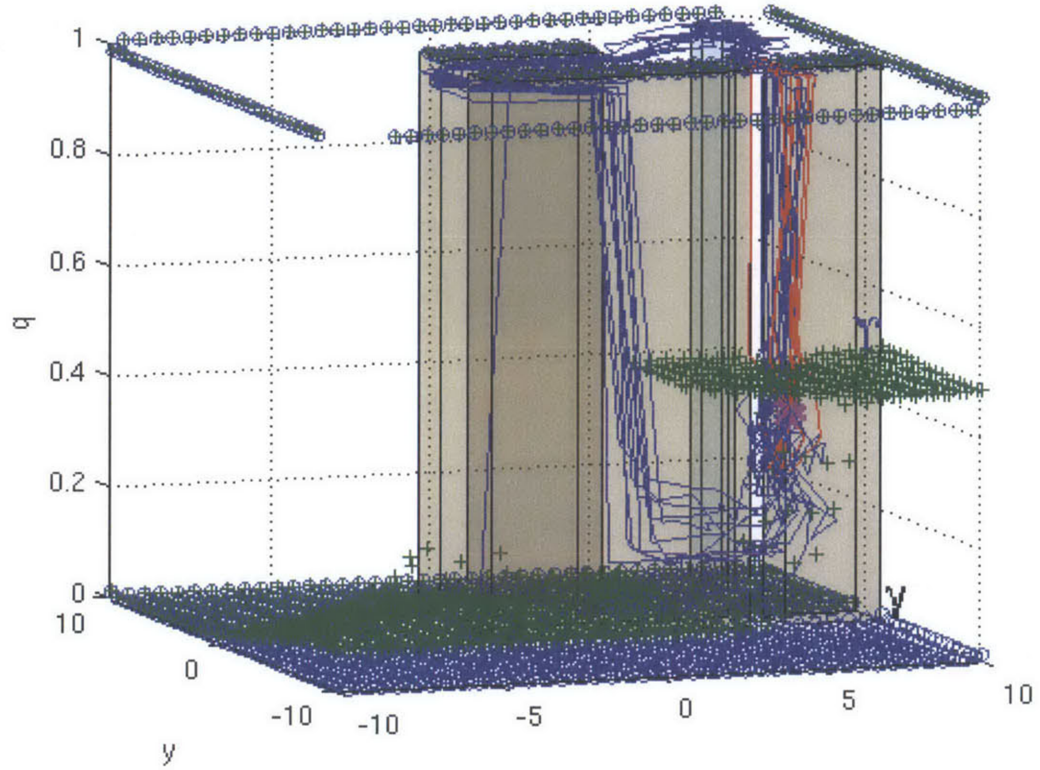
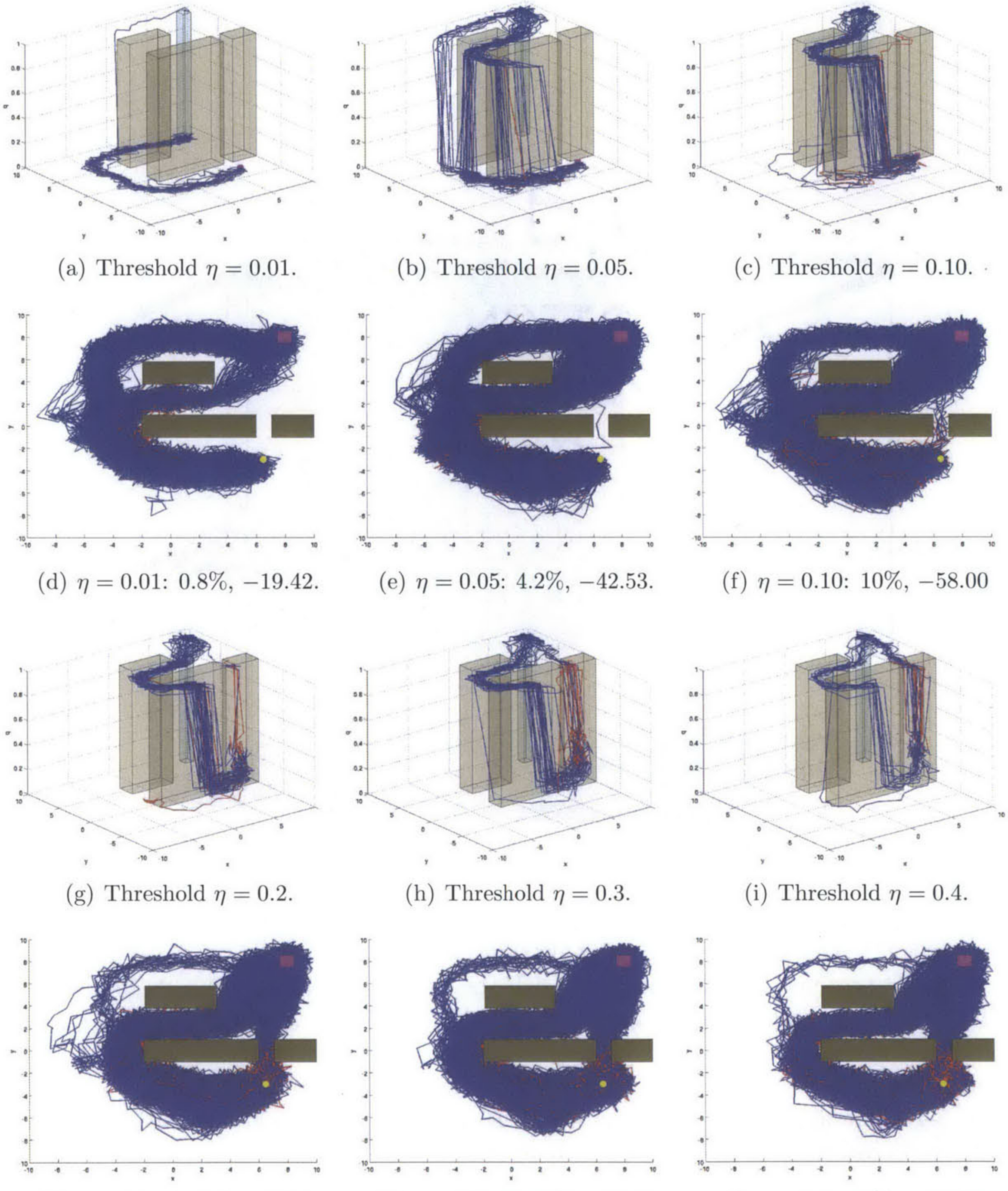


Figure 5-11: An example of controlled trajectories using boundary values for Fig. 5-4. The system starts from $(6.5, -3)$ with the failure-probability threshold $\eta = 0.4$. The martingale state varies along controlled trajectories as a random parameter in a randomized control policy. When the martingale state is above Υ, the system follows a deterministic control policy obtained from the unconstrained problem. As seen in Fig. 5-13, the algorithm is able to keep the failure ratio in 5000 executions around 0.40 as dictated by the choice of $\eta = 0.40$ at time 0.



(a) Threshold $\eta = 0.01$.

(b) Threshold $\eta = 0.05$.

(c) Threshold $\eta = 0.10$.

(d) $\eta = 0.01$: 0.8%, -19.42.

(e) $\eta = 0.05$: 4.2%, -42.53.

(f) $\eta = 0.10$: 10%, -58.00

(g) Threshold $\eta = 0.2$.

(h) Threshold $\eta = 0.3$.

(i) Threshold $\eta = 0.4$.

(j) $\eta = 0.2$: 15.6%, -65.81.

(k) $\eta = 0.3$: 28.19%, -76.80.

(l) $\eta = 0.4$: 40%, -115.59.

Figure 5-12: Trajectories after 5000 iterations starting from $(6.5, -3)$. In Figs. 5-12(a)-5-12(c) and Figs. 5-12(g)-5-12(i), we show 50 trajectories resulting from a policy induced by J_{4000} with different collision-probability thresholds ($\eta = 0.01, 0.05, 0.10, 0.20, 0.30, 0.40$). In Figs. 5-12(d)-5-12(f) and Figs. 5-12(j)-5-12(l), we show 5000 corresponding trajectories in the original state space S with *simulated collision probabilities* and *average costs* in their captions. Trajectories that reach the goal region are plotted in blue, and trajectories that hit obstacles are plotted in red.

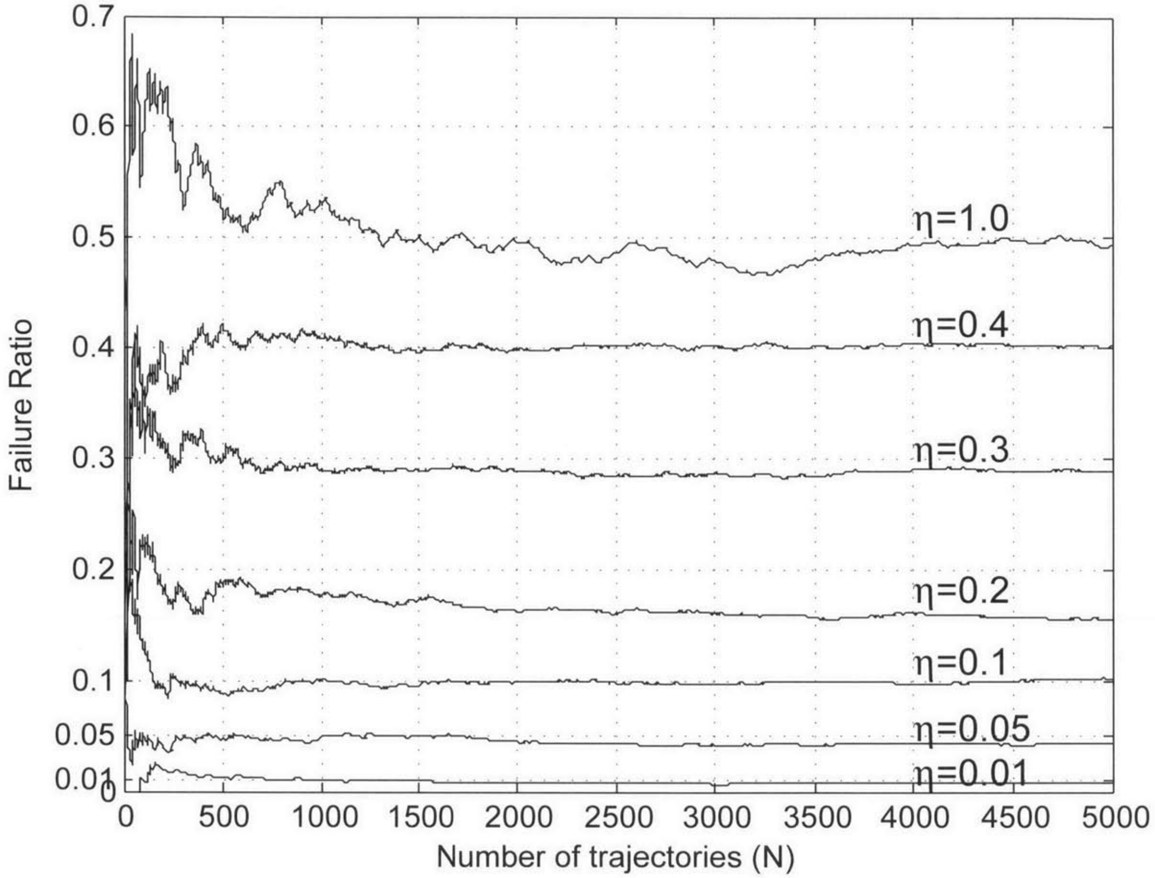


Figure 5-13: Failure ratios for the first N trajectories ($1 \leq N \leq 5000$) starting from $(6.5, -3)$ with different values of η. These failure ratios follow very closely the values of η, which indicates that the iMDP algorithm is able to provide solutions that are probabilistically sound.

Chapter 6

Conclusions

Sampling-based algorithms have received much attention from the robotics community as a randomized approach to solve the fundamental deterministic robot motion planning. The motivation of this thesis is to address the robot motion planning in uncertain environments. This problem is formulated abstractly as a stochastic optimal control problem. The formulation is also general enough for a wide range of potential applications in biology, healthcare, and management.

Therefore, in this thesis, we have introduced a set of new sampling-based algorithms for solving a general class of continuous-time continuous-space stochastic optimal control problems in the presence of complex risk constraints. In the following, we will first summarize the algorithms and results developed in this thesis, and subsequently present possible directions for future research.

6.1 Summary

The main contribution of this thesis is a new computationally-efficient sampling-based algorithm called the incremental Markov Decision Process (iMDP) algorithm that provides asymptotically-optimal solutions to continuous time and space stochastic control problems.

The iMDP algorithm constructs a sequence of approximating finite-state Markov Decision Processes (MDPs) that consistently approximates the original continuous-time stochastic dynamics and solves the optimal control problem in an incremental manner. Using the rapidly-exploring sampling technique to sample in the state space, iMDP forms the structures of finite-state MDPs randomly over iterations. Control sets for states in these MDPs are constructed or sampled properly in the control space. The finite models serve as incrementally refined models of the original problem. More precisely, the connected random graph structures of Markov chains on MDPs explore well the original state space. To have consistent approximation, only the mean and covariance of displacement per step along a Markov chain under any control are required to be close enough to those of the original dynamics. Consequently, the distributions of approximating trajectories and control processes returned from these finite models approximate arbitrarily well the distributions of optimal trajectories and optimal control processes of the original problem.

The iMDP algorithm approximates the optimal cost-to-go function using the asynchronous Bellman value iteration procedure such that computation in the current iteration is inherited from the previous iterations. We show that the time complexity per iteration grows as the product of fractional power and polylogarithmic time while the space complexity grows linearly over iterations.

There are two main advantages to use the iMDP algorithm for solving stochastic optimal control problems. First, the iMDP algorithm provides a method to compute optimal control policies without the need to derive and characterize viscosity solutions of the associated HJB equations. Second, the algorithm is suitable for various online robotics applications without *a priori* discretization of the state space.

Risk management has always been an important part of stochastic optimal control problems to guarantee safety during the execution of control policies. We consider two types of risk constraints in this thesis. The first type of risk constraints called bounded trajectory performance that has the same integration structure as the objective function with different cost rate, terminal cost functions and discount factors. We enforce this type of constraint for all sub-trajectories along the controlled process. The iMDP algorithm has been extended to provide probabilistically-sound and asymptotically optimal control policies for this class of constrained stochastic control problems. The returned policies from the original iMDP and this extended version are deterministic function of states.

The second type of risk constraints is called bounded probability of failure, which is enforced for particular initial states. We have introduced the martingale approach to handle probability constraints on the terminal states. The martingale approach transforms the probability-constrained problem into an equivalent stochastic target problem with the augmented state and control spaces. The boundary conditions for the transformed problem is, however, unspecified. We have presented a new extended version of the iMDP algorithm that incrementally computes the boundary values and any-time feedback control policies for the transformed problem. The returned policies can be considered as randomized policies in the original state space. Effectively, the extended iMDP algorithm provides probabilistically-sound and asymptotically-optimal control policies for the class of stochastic control problems with bounded failure-probability constraints.

6.2 Future Directions

In this sections, we present some directions for future research on related problems.

6.2.1 Partially-observable states

In several systems, true states are not available during the controlled process. Instead, there are sensors to provide noisy measurements of unknown states. Controlling systems in these situations leads to a class of stochastic optimal control problems with imperfect state information, known as Partially Observable Markov Decision

Processes (POMDPs). Although POMDPs are fundamentally more challenging than the problem that is studied in this paper, our approach differentiates itself from existing sampling-based POMDP solvers (see, e.g., [146]) with its incremental nature and computationally-efficient search. Hence, the research presented in this paper opens a new alley to handle POMDPs.

Recent research by Chaudhari et al. [147, 148] has explored this direction for the problem of state estimation and POMDPs. In [148], the authors use an approximating sequence of discrete time finite-state POMDPs to approximate continuous POMDPs such that the optimal cost function and control policies for these POMDP approximations converge almost surely to their counterparts for the underlying continuous systems in the limit. For each POMDP approximation, the authors use an existing POMDP solver, SARSOP [146], to obtain a policy for the POMDP approximation. However, SARSOP still encounters major computational challenges for practical systems in high dimensional state spaces. As a result, providing efficient approximate solutions to POMDPs is still an open research problem.

One possible research direction is to provide *incremental* computation of policies without fully solving each finite-state POMDP using SARSOP. This is also the key idea behind the iMDP algorithm. Another research idea is to combine results in information theory and control theory such that we can better utilize sensor data to design better approximating structures for the continuous time and space POMDPs.

6.2.2 Error bounds for anytime policies

Although anytime policies in this thesis are asymptotically optimal, we have not investigated the error bounds of the cost-to-go function under these policies in comparison to the optimal cost-to-go function. Estimates of error bounds would provide better understanding in the quality of anytime policies. The upper bounds on the cost-to-go function can be found by simulating the returned policies. Estimating the lower bounds is more challenging and is an active research topic.

One possible approach called *information relaxations* can be used to find the lower bounds (see, e.g. [149, 150] and references therein). In this approach, we relax the nonanticipativity constraints that require decisions to depend only on the information available at the time a decision is made and impose a "penalty" that punishes violations of nonanticipativity. In many cases, the relaxed version of the problem is simple to solve and provides the lower bounds. We suggest a future research direction that incorporates information relaxations into the sampling-based iMDP algorithm to provide useful anytime error bounds.

6.2.3 Investigation of new noise models

Noise can be driven by not only Brownian processes but also jump processes so that the controlled process has the form:

$$x(t) = x(0) + \int_0^t f(x(\tau), u(\tau))d\tau + \int_0^\tau F(x(\tau), u(\tau)dw(\tau) + J(t),$$

where the term $J(t)$ produces the jumps. To characterize the jump term, we would like to specify the probability that a jump occurs in any small time interval and the distribution of any resulting jumps as the function of the past history process. Between jumps, the term $J(t)$ is constant.

The Markov chain approximation method can be extended to handle the stochastic process with jumps (see Chapter 5 of [43]). The local consistency conditions now include the approximation for jump intensities during holding times. As a result, convergence results will follow. We would like to extend the iMDP algorithm to provide incremental computation of anytime policies for this class of stochastic dynamics.

6.2.4 Logic constraints

In reality, complex systems obey not only physical rules but also logical rules set by authorities and operators that specify valid sequences of allowed operations. Such constraints are useful to enable self-driving cars to follow traffic law or steerable medical needles to follow safety guidelines. Temporal logic as a formal high level language can describe succinctly these constraints.

Current research such as [151–154] has investigated similar logic constraints for the robot motion planning problem in deterministic environments. The main idea is to construct suitable approximating discrete structures for logic constraints that represent well the original logical rules in the continuous state space.

However, controlling complex systems with temporal logic constraints in the presence of disturbances is a challenging unexplored problem. We would like to extend the sampling-based approach presented in this thesis to incorporate such logic constraints. In particular, one promising future research direction is to investigate suitable approximating structures for these logic constraints such that they can be combined with the approximating MDPs structure in an efficient and effective way.

6.2.5 Exploiting dynamics of mechanical systems

This thesis has focused on general system dynamics. For robotics applications, we often deal with nonlinear dynamics with special properties such as underactuation [155] and differential flatness [156, 157]. Exploiting these properties to design optimal control policies would provide higher performance in many situations. Designing new versions of the iMDP algorithm that incorporate directly these properties is left for future investigation.

6.2.6 Parallel implementation

As our considered stochastic optimal control problems become more complex due to both risk constraints and logic constraints, despite low theoretical time complexity per iteration guarantees, the actual running time to compute anytime solutions for such problems would increase significantly. Therefore, parallel implementation for iMDP-like algorithms would be highly desirable to obtain fast running time. We note that the algorithms presented in this thesis are highly parallelizable by design. An interesting research direction is to combine parallelization and the interdependence of primitive procedures in the iMDP algorithm to speed up its running time. This direction is similar to the ideas proposed by Bialkowski [97] for RRT-like algorithms.

6.2.7 Collaborative multi-agent systems

We can further consider a team of separate and independent agents collaborating to optimize a common objective function in uncertain environments. Each agent can compute a policy in their explored state space and is able to communicate its computed policy and intension with other agents through possibly bandwidth-limited and unreliable networks.

One possible direction for this problem is to extend the iMDP algorithm so that each agent constructs its own approximating data structures in its interested regions of the state space. Agents are coordinated to communicate these approximating data structures with each other, and they can further refine their own approximating data structures based on received information. Designing a coordination plan that enables each agent to compute good approximations of an optimal control policy while minimizing the amount of data transfered is an interesting research question to answer.

6.2.8 Stochastic games

In stochastic games, we have several agents each operating independently and strategically to optimize their own objective functions in the presence of uncertainties. Each agent can observe other agents' trajectories to compute their decision at any moment. These problems form an interesting and challenging class of stochastic optimal control problems.

Recent works develop the weak dynamic programming principle for zero-sum games in continuous-time [158, 159] and further derive Partial Differential Equations for this sub-class of games. Developing incremental policies for each agent that are consistent with their observations and initial requirements is an open research question. We suggest an approach that is similar to the martingale approach presented in this work as one possible direction.

6.2.9 Exploring applications in management and finance

The formulation considered in this thesis is fairly abstract and can find applications in many areas such as mathematical economics and finance. Examples of these problems are optimal dynamic contract design [1], optimal hedging in the presence of proportional transaction costs, and liquidation with a target costs constraint [24]. Applying the ideas in the iMDP algorithm to design algorithms for these applications is an interesting research direction to pursue.

Bibliography

[1] L. Ljungqvist and T. J. Sargent, *Recursive macroeconomic theory.* The MIT press, 2004.

[2] N. L. Stokey, R. E. Lucas, and E. C. Prescott, *Recursive methods in economic dynamics.* Harvard University Press, 1989.

[3] H. Pham, *Continuous-time stochastic control and optimization with financial applications.* Springer, 2009, vol. 61.

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

[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. Siméon, 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] 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.

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

[9] Y. Kuwata, M. Pavone, and J. Balaram, "A risk-constrained multi-stage decision making approach to the architectural analysis of planetary missions," in *CDC*, 2012, pp. 2102–2109.

[10] 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.

[11] M. Pavone, S. L. Smith, E. Frazzoli, and D. Rus, "Robotic load balancing for mobility-on-demand systems," *I. J. Robotic Res.*, vol. 31, no. 7, pp. 839 854, 2012.

[12] R. P. Anderson, E. Bakolas, D. Milutinovic, and P. Tsiotras, "The markov-dubins problem in the presence of a stochastic drift field," in *CDC*, 2012, pp. 130–135.

[13] R. Bellman, *Dynamic Programming*, 1st ed. Princeton, NJ, USA: Princeton University Press, 1957.

[14] W. H. Fleming, "Optimal continuous-parameter stochastic control," *SIAM review*, vol. 11, no. 4, pp. 470–509, 1969.

[15] W. H. Fleming and R. W. Rishel, *Deterministic and stochastic optimal control.* Springer New York, 1975, vol. 1.

[16] M. H. Davis, *Martingale methods in stochastic control.* Springer, 1979.

[17] W. H. Fleming and H. M. Soner, *Controlled Markov processes and viscosity solutions.* Springer New York, 2006, vol. 25.

[18] J. Yong and X. Y. Zhou, *Stochastic Controls: Hamiltonian Systems and HJB Equations*, 1st ed. Springer, June 1999.

[19] V. Beneš, "Full bang to reduce predicted miss is optimal," *SIAM Journal on Control and Optimization*, vol. 14, no. 1, pp. 62–84, 1976.

[20] M. G. Crandall and P.-L. Lions, "Viscosity solutions of hamilton-jacobi equations," *Transactions of the American Mathematical Society*, vol. 277, no. 1, pp. 1–42, 1983.

[21] B. Bouchard and N. Touzi, "Weak dynamic programming principle for viscosity solutions," *SIAM Journal on Control and Optimization*, vol. 49, no. 3, pp. 948 962, 2011.

[22] B. Bouchard, R. Elie, and C. Imbert, "Optimal control under stochastic target constraints," *SIAM Journal on Control and Optimization*, vol. 48, no. 5, pp. 3501–3531, 2010.

[23] B. Bouchard, R. Elie, and N. Touzi, "Stochastic target problems with controlled loss," *SIAM Journal on Control and Optimization*, vol. 48, no. 5, pp. 3123 3150, 2009.

[24] N. Touzi and A. Tourin, *Optimal stochastic control, stochastic target problems, and backward SDE.* Springer, 2013, vol. 29.

[25] B. Bouchard and M. Nutz, "Weak dynamic programming for generalized state constraints," *SIAM Journal on Control and Optimization*, vol. 50, no. 6, pp. 3344–3373, 2012.

[26] H. M. Soner and N. Touzi, "Stochastic target problems, dynamic programming, and viscosity solutions," *SIAM Journal on Control and Optimization*, vol. 41, no. 2, pp. 404–424, 2002.

[27] 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.

[28] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of markov decision processes," *Mathematics of operations research*, vol. 12, no. 3, pp. 441–450, 1987.

[29] M. L. Littman, T. L. Dean, and L. P. Kaelbling, "On the complexity of solving markov decision problems," in *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence.* Morgan Kaufmann Publishers Inc., 1995, pp. 394–402.

[30] M. L. Littman, "Probabilistic propositional planning: Representations and complexity," in *Proceedings of the National Conference on Artificial Intelligence.* JOHN WILEY & SONS LTD, 1997, pp. 748–754.

[31] M. Mundhenk, J. Goldsmith, C. Lusena, and E. Allender, "Encyclopaedia of complexity results for finite-horizon markov decision process problems," *Certer for Discrete Mathematics & Theoretical Computer Science*, 1997.

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

[33] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Two Volume Set*, 2nd ed. Athena Scientific, 2001.

[34] R. C. Merton, *Optimum consumption and portfolio rules in a continuous-time model.* MIT, 1970.

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

[36] 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.

[37] 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.

[38] A. Budhiraja and K. Ross, "Convergent numerical scheme for singular stochastic control with state constraints in a portfolio selection problem," *SIAM Journal on Control and Optimization*, vol. 45, no. 6, pp. 2169–2206, 2007.

[39] 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.

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

[41] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.

[42] ——, *Neuro-Dynamic Programming (Optimization and Neural Computation Series, 3).* Athena Scientific, May 1996.

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

[44] H. J. Kushner and H. Joseph, *Probability methods for approximations in stochastic control and for elliptic equations.* Academic Press New York, 1977, vol. 129.

[45] P. Billingsley, *Convergence of probability measures.* Wiley-Interscience, 2009, vol. 493.

[46] ——, *Probability and measure.* Wiley, 2012, vol. 939.

[47] A. V. Skorokhod, "Limit theorems for stochastic processes," *Theory of Probability & Its Applications*, vol. 1, no. 3, pp. 261–290, 1956.

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

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

[50] 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.

[51] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Ames, IA, Tech. Rep. 98-11, Oct. 1998.

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

[53] J.-C. Latombe, *Robot Motion Planning.* Norwell, MA, USA: Kluwer Academic Publishers, 1991.

[54] A. Bhatia and E. Frazzoli, *Incremental search methods for reachability analysis of continuous and hybrid systems.* Springer, 2004.

[55] J. Cortés, L. Jaillet, and T. Siméon, "Molecular disassembly with rrt-like algorithms," in *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007, pp. 3301–3306.

[56] Y. Liu and N. I. Badler, "Real-time reach planning for animated characters using hardware acceleration," in *Computer Animation and Social Agents, 2003. 16th International Conference on*. IEEE, 2003, pp. 86–93.

[57] P. Finn and L. E. Kavraki, "Computational approaches to drug design," *Algorithmica*, vol. 25, no. 2-3, pp. 347–371, 1999.

[58] J.-C. Latombe, "Motion planning: A journey of robots, molecules, digital actors, and other artifacts," *The International Journal of Robotics Research*, vol. 18, no. 11, pp. 1119–1128, 1999.

[59] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[60] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, "Principles of robot motion: theory, algorithms, and implementations. 2005," *MIT Press, Boston*.

[61] J. H. Reif, "Complexity of the movers problem and generalizations extended abstract," in *Proceedings of the 20th Annual IEEE Conference on Foundations of Computer Science*, 1979, pp. 421–427.

[62] J. F. Canny, *The Complexity of Robot Motion Planning*. Cambridge, MA, USA: MIT Press, 1988.

[63] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, 1979.

[64] J. T. Schwartz and M. Sharir, "On the piano movers problem. ii. general techniques for computing topological properties of real algebraic manifolds," *Advances in applied Mathematics*, vol. 4, no. 3, pp. 298–351, 1983.

[65] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," in *Foundations of Computer Science, 1987., 28th Annual Symposium on*. IEEE, 1987, pp. 49–60.

[66] R. A. Brooks and T. Lozano-Perez, "A subdivision algorithm in configuration space for findpath with rotation," *Systems, Man and Cybernetics, IEEE Transactions on*, no. 2, pp. 224–233, 1985.

[67] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[68] S. S. Ge and Y. Cui, "Dynamic motion planning for mobile robots using potential field method," *Autonomous Robots*, vol. 13, no. 3, pp. 207–222, 2002.

[69] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on.* IEEE, 1991, pp. 1398–1404.

[70] L. Kavraki and J.-C. Latombe, "Randomized preprocessing of configuration for fast path planning," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on.* IEEE, 1994, pp. 2138–2145.

[71] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[72] S. R. Lindemann and S. M. LaValle, "Current issues in sampling-based motion planning," in *Robotics Research.* Springer, 2005, pp. 36–54.

[73] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in belief space by factoring the covariance," *The International Journal of Robotics Research*, vol. 28, no. 11-12, pp. 1448–1465, 2009.

[74] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "Lqr-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[75] B. D. Luders, S. Karaman, E. Frazzoli, and J. P. How, "Bounds on tracking error using closed-loop rapidly-exploring random trees," in *American Control Conference (ACC), 2010.* IEEE, 2010, pp. 5406–5412.

[76] J. Barraquand, L. Kavraki, R. Motwani, J.-C. Latombe, T.-Y. Li, and P. Raghavan, "A random sampling scheme for path planning," in *Robotics Research.* Springer, 2000, pp. 249–264.

[77] L. E. Kavraki, M. N. Kolountzakis, and J.-C. Latombe, "Analysis of probabilistic roadmaps for path planning," *Robotics and Automation, IEEE Transactions on*, vol. 14, no. 1, pp. 166–171, 1998.

[78] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 116–129, 2002.

[79] A. M. Ladd and L. E. Kavraki, "Measure theoretic analysis of probabilistic path planning," *Robotics and Automation, IEEE Transactions on*, vol. 20, no. 2, pp. 229–242, 2004.

[80] E. Koyuncu, N. K. Ure, and G. Inalhan, "Integration of path/maneuver planning in complex environments for agile maneuvering ucavs," in *Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009.* Springer, 2010, pp. 143–170.

[81] D. Berenson, J. Kuffner, and H. Choset, "An optimization approach to planning for mobile manipulation," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on.* IEEE, 2008, pp. 1187–1192.

[82] A. Yershova and S. M. LaValle, "Motion planning for highly constrained spaces," in *Robot Motion and Control 2009.* Springer, 2009, pp. 297–306.

[83] M. Stilman, J.-U. Schamburek, J. Kuffner, and T. Asfour, "Manipulation planning among movable obstacles," in *Robotics and Automation, 2007 IEEE International Conference on.* IEEE, 2007, pp. 3327–3332.

[84] J. J. Kuffner Jr and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 995–1001.

[85] M. S. Branicky, M. M. Curtiss, J. A. Levine, and S. B. Morgan, "Rrts for nonlinear, discrete, and hybrid planning and control," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 1. IEEE, 2003, pp. 657–663.

[86] M. S. Branicky, M. M. Curtiss, J. Levine, and S. Morgan, "Sampling-based planning, control and verification of hybrid systems," *IEE Proceedings-Control Theory and Applications*, vol. 153, no. 5, pp. 575–590, 2006.

[87] M. Zucker, J. Kuffner, and M. Branicky, "Multipartite rrts for rapid replanning in dynamic environments," in *Robotics and Automation, 2007 IEEE International Conference on.* IEEE, 2007, pp. 1603–1609.

[88] C. Urmson and R. Simmons, "Approaches for heuristically biasing rrt growth," in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 2. IEEE, 2003, pp. 1178–1183.

[89] D. Ferguson and A. Stentz, "Anytime rrts," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on.* IEEE, 2006, pp. 5369–5375.

[90] N. A. Wedge and M. S. Branicky, "On heavy-tailed runtimes and restarts in rapidly-exploring random trees," in *Twenty-Third AAAI Conference on Artificial Intelligence*, 2008, pp. 127–133.

[91] L. Jaillet, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *Robotics, IEEE Transactions on*, vol. 26, no. 4, pp. 635–646, 2010.

[92] D. Berenson, T. Siméon, and S. S. Srinivasa, "Addressing cost-space chasms in manipulation planning," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on.* IEEE, 2011, pp. 4561–4568.

[93] 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.

[94] J. Bruce and M. Veloso, "Real-time randomized path planning for robot navigation," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 3. IEEE, 2002, pp. 2383–2388.

[95] A. Shkolnik, M. Levashov, I. R. Manchester, and R. Tedrake, "Bounding on rough terrain with the littledog robot," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 192–215, 2011.

[96] S. Teller, M. R. Walter, M. Antone, A. Correa, R. Davis, L. Fletcher, E. Frazzoli, J. Glass, J. P. How, A. S. Huang, *et al.*, "A voice-commandable robotic forklift working alongside humans in minimally-prepared outdoor environments," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 526–533.

[97] B. J., "Optimizations for sampling-based motion planning algorithms," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge,MA,USA, Dec. 2013.

[98] M. Otte, J. Bialkowski, and E. Frazzoli, "Any-com collision checking: Sharing certificates in decentralized multi-robot teams," in *IEEE Int. Conf. on Robotics and Automation*, 2014, to appear.

[99] J. Bialkowski, M. Otte, S. Karaman, and F. E., "Efficient collision checking in sampling-based motion planning," in *Algorithmic Foundations of Robotics X*, ser. Springer Tracts in Advanced Robotics. Springer Berlin / Heidelberg, 2013, vol. 69.

[100] J. Bialkowski, M. W. Otte, and E. Frazzoli, "Free-configuration biased sampling for motion planning," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[101] E. N. Gilbert, "Random plane networks," *Journal of the Society for Industrial & Applied Mathematics*, vol. 9, no. 4, pp. 533–543, 1961.

[102] M. Penrose, *Random geometric graphs*. Oxford University Press Oxford, 2003, vol. 5.

[103] P. Balister, B. Bollobás, A. Sarkar, and M. Walters, "Connectivity of random k-nearest-neighbour graphs.(english summary)," *Adv. in Appl. Probab*, vol. 37, no. 1, pp. 1–24, 2005.

[104] B. Bollobás, *Random graphs*. Cambridge university press, 2001, vol. 73.

[105] P. Gupta and P. R. Kumar, "Critical power for asymptotic connectivity in wireless networks," in *Stochastic analysis, control, optimization and applications*. Springer, 1998, pp. 547 566.

[106] ——, "The capacity of wireless networks," *Information Theory, IEEE Transactions on*, vol. 46, no. 2, pp. 388–404, 2000.

[107] E. A. Feinberg, E. A. Feinberg, A. Shwartz, and A. Shwartz, "Constrained markov decision models with weighted discounted rewards," *Math. of Operations Research*, vol. 20, pp. 302–320, 1993.

[108] E. Feinberg and A. Shwartz, "Constrained dynamic programming with two discount factors: applications and an algorithm," *Automatic Control, IEEE Transactions on*, vol. 44, no. 3, pp. 628–631, Mar 1999.

[109] 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.

[110] 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.

[111] Y. L. Chow and M. Pavone, "Stochastic optimal control with dynamic, time-consistent risk constraints," in *American Control Conference (ACC), 2012*. IEEE, 2012. Submitted.

[112] B. D. Luders, S. Karaman, and J. P. How, "Robust sampling-based motion planning with asymptotic optimality guarantees," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, Boston, MA, August 2013.

[113] B. Luders, M. Kothari, and J. P. How, "Chance constrained RRT for probabilistic robustness to environmental uncertainty," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, Toronto, Canada, August 2010, (AIAA-2010-8160).

[114] 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.

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

[116] L. Blackmore, H. Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *in Proceedings of the American Control Conference*, 2006.

[117] M. Ono and B. C. Williams, "Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint," in *CDC*, 2008, pp. 3427–3432.

[118] G. S. Aoude, B. D. Luders, J. M. Joseph, N. Roy, and J. P. How, "Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns," *Autonomous Robots*, vol. 35, no. 1, pp. 51–76, 2013.

[119] R. C. Chen and G. L. Blankenship, "Dynamic programming equations for discounted constrained stochastic control," *Automatic Control, IEEE Transactions on*, vol. 49, no. 5, pp. 699–709, 2004.

[120] A. Piunovskiy, "Dynamic programming in constrained markov decision processes," *Control and Cybernetics*, vol. 35, no. 3, p. 645, 2006.

[121] S. Mannor and J. Tsitsiklis, "Mean-variance optimization in markov decision processes," *arXiv preprint arXiv:1104.5601*, 2011.

[122] P. Huang, D. A. Iancu, M. Petrik, and D. Subramanian, "The price of dynamic inconsistency for distortion risk measures," *arXiv preprint arXiv:1106.6102*, 2011.

[123] A. Ruszczyński and A. Shapiro, "Optimization of risk measures," in *Probabilistic and randomized methods for design under uncertainty.* Springer, 2006, pp. 119 157.

[124] —— , "Conditional risk mappings," *Mathematics of Operations Research*, vol. 31, no. 3, pp. 544–561, 2006.

[125] B. Rudloff, A. Street, and D. Valladao, "Time consistency and risk averse dynamic decision models: Interpretation and practical consequences," *Internal Research Reports*, vol. 17, 2011.

[126] H. M. Soner and N. Touzi, "Dynamic programming for stochastic target problems and geometric flows," *Journal of the European Mathematical Society*, vol. 4, no. 3, pp. 201 236–236, Sept. 2002.

[127] B. Bouchard and T. N. Vu, "The obstacle version of the geometric dynamic programming principle: Application to the pricing of american options under constraints," *Applied Mathematics and Optimization*, vol. 61, no. 2, pp. 235–265, 2010.

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

[129] B. Oksendal, *Stochastic differential equations (3rd ed.): an introduction with applications.* New York, NY, USA: Springer-Verlag New York, Inc., 1992.

[130] "Review of probability theory," http://terrytao.wordpress.com/2010/01/01/254a-notes-0-a-review-of-probability-theory/, accessed: 2014-03-20.

[131] H. M. Soner and N. Touzi, "Dynamic programming for stochastic target problems and geometric flows," *Journal of the European Mathematical Society*, vol. 4, no. 3, pp. 201–236–236, Sept. 2002.

[132] D. Bertsekas and J. Tsitsiklis, *Introduction To Probability*, ser. Athena Scientific books. Athena Scientific, 2002.

[133] B. Bollobas and O. Riordan, *Percolation.* Cambridge University Press, 2006.

[134] F. Xue and P. R. Kumar, "The number of neighbors needed for connectivity of wireless networks," *Wirel. Netw.*, vol. 10, no. 2, pp. 169–181, Mar. 2004.

[135] D. Stoyan, W. Kendall, and J. Mecke, *Stochastic geometry and its applications.* Wiley series in probability and statistics, 1995.

[136] M. Maier, M. Hein, and U. von Luxburg, "Cluster identification in nearest-neighbor graphs." in *ALT*, ser. Lecture Notes in Computer Science, M. Hutter, R. A. Servedio, and E. Takimoto, Eds., vol. 4754. Springer, 2007, pp. 196–210.

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

[138] ——, "An incremental sampling-based algorithm for stochastic optimal control," *arXiv:1202.5544v1 [cs.RO]*, 2012.

[139] L. C. Evans, *Partial Differential Equations (Graduate Studies in Mathematics, V. 19) GSM/19.* American Mathematical Society, June 1998.

[140] J. L. Menaldi, "Some estimates for finite difference approximations," *SIAM J. on Control and Optimization*, vol. 27, pp. 579–607, 1989.

[141] P. Dupuis and M. R. James, "Rates of convergence for approximation schemes in optimal control," *SIAM J. Control Optim.*, vol. 36, pp. 719–741, March 1998.

[142] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, Mar. 2004.

[143] G. R. Grimmett and D. R. Stirzaker, *Probability and Random Processes*, 3rd ed. Oxford University Press, USA, Aug. 2001.

[144] V. A. Huynh and E. Frazzoli, "Probabilistically-sound and asymptotically-optimal algorithm for stochastic control with trajectory constraints," in *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on.* IEEE, 2012, pp. 1486–1493.

[145] V. A. Huynh, L. Kogan, and E. Frazzoli, "A martingale approach and time-consistent sampling-based algorithms for risk management in stochastic optimal control," *CoRR*, vol. abs/1312.7602, 2013.

[146] H. Kurniawati, D. Hsu, and W. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Proc. Robotics: Science and Systems*, 2008.

[147] P. Chaudhari, S. Karaman, and E. Frazzoli, "Sampling-based algorithm for filtering using markov chain approximations." in *CDC*. IEEE, 2012, pp. 5972–5978.

[148] P. Chaudhari, S. Karaman, D. Hsu, and E. Frazzoli, "Sampling-based algorithms for Continuous-time POMDPs," in *Proc. American Control Conf.*, 2013.

[149] D. B. Brown and J. E. Smith, "Dynamic portfolio optimization with transaction costs: Heuristics and dual bounds," *Management Science*, vol. 57, no. 10, pp. 1752–1770, 2011.

[150] D. B. Brown, J. E. Smith, and P. Sun, "Information relaxations and duality in stochastic dynamic programs," *Operations Research*, vol. 58, no. 4-Part-1, pp. 785–801, 2010.

[151] J. Tumova, L. I. Reyes Castro, S. Karaman, E. Frazzoli, and D. Rus, "Minimum-violation LTL planning with conflicting specifications," in *American Control Conference*, 2013, pp. 200–205.

[152] J. Tumova, S. Karaman, G. Hall, E. Frazzoli, and D. Rus, "Least-violating control strategy synthesis with safety rules," in *Hybrid Systems: Computation and Control*, 2013.

[153] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning with deterministic μ-calculus specifications," in *Proc. American Control Conf.*, 2012.

[154] T. Wongpiromsarn and E. Frazzoli, "Control of probabilistic systems under dynamic, partially known environments with temporal logic specifications," in *IEEE Conf. on Decision and Control*, Maui, HI, 2012, pp. 7644–7651.

[155] R. Olfati-Saber, "Nonlinear control of underactuated mechanical systems with application to robotics and aerospace vehicles," Ph.D. dissertation, 2001, aAI0803036.

[156] S. C. Peters, E. Frazzoli, and K. Iagnemma, "Differential flatness of a front-steered vehicle with tire force control," in *IROS*, 2011, pp. 298–304.

[157] J. Jeon, R. Cowlagi, S. Peters, S. Karaman, E. Frazzoli, P. Tsiotras, and K. Iagnemma, "Optimal motion planning with the half-car dynamical model for autonomous high-speed driving," in *American Control Conference (ACC) (to appear)*, 2013.

[158] P. Vitória, "A weak dynamic programming principle for zero-sum stochastic differential games," *Master degree thesis, Universidade Técnica de Lisboa*, 2010.

[159] B. Erhan and Y. Song, "A weak dynamic programming principle for zero-sum stochastic differential games with unbounded controls," *arXiv preprint arXiv:1210.2788*, 2013.