Robot Motion Planning with Uncertainty and Urgency

Brian Hou

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington
2023

Reading Committee:
Siddhartha Srinivasa, Chair
Byron Boots
Sanjiban Choudhury

Program Authorized to Offer Degree:
Computer Science & Engineering

University of Washington

Abstract

Robot Motion Planning with Uncertainty and Urgency

Brian Hou

Chair of the Supervisory Committee:
Siddhartha Srinivasa
Paul G. Allen School of Computer Science & Engineering

As robots are introduced to a wider variety of real-world domains—factories, roads, and homes—they must be able to reliably operate with incomplete knowledge of the cluttered-but-structured environment. This dissertation considers the problem of *motion planning with uncertainty*, where a robot navigates to a goal without knowing the environment's exact obstacle geometry. How can uncertainty be efficiently incorporated into decision-making, without ballooning planning times on computationally-constrained systems and paralyzing robots into indecision? How can algorithms leverage the *structure* of uncertainty to efficiently plan high-quality paths?

This dissertation proposes Bayesian strategies for integrating uncertainty throughout the sampling-based motion planning framework. We formalize uncertainty with an informative posterior distribution over latent environment parameters. In this component, roboticists express their domain expertise about a robot's environment, sensor suite, and interactions between the two. Bayesian planning algorithms aim to navigate the exploration-exploitation tradeoff with respect to this distribution of environments that the robot anticipates seeing at test time.

We develop efficient Bayesian search algorithms motivated by *regret minimization*. This objective captures the urgency of a planner's sequential decision-making process by comparing with the optimal decision-maker at each iteration. The cumulative difference, or regret, penalizes suboptimality at each iteration as well as the time expended to reduce that suboptimality. We demonstrate that algorithms based on posterior sampling are effective for Bayesian anytime lazy motion planning and Bayesian dynamic motion planning.

We propose a *variational inference* algorithm for roadmap optimization, which aims to match the distribution of roadmap samples to the target distribution of collision-free states. We demonstrate that optimized sparse roadmaps concisely approximate the uncertain environment and can be searched more efficiently than conventional uniform or low-discrepancy dense roadmaps.

Abstract

As robots are introduced to a wider variety of real-world domains—factories, roads, and homes—they must be able to reliably operate with incomplete knowledge of the cluttered-but-structured environment. This dissertation considers the problem of *motion planning with uncertainty*, where a robot navigates to a goal without knowing the environment's exact obstacle geometry. How can uncertainty be efficiently incorporated into decision-making, without ballooning planning times on computationally-constrained systems and paralyzing robots into indecision? How can algorithms leverage the *structure* of uncertainty to efficiently plan high-quality paths?

This dissertation proposes Bayesian strategies for integrating uncertainty throughout the sampling-based motion planning framework. We formalize uncertainty with an informative posterior distribution over latent environment parameters. In this component, roboticists express their domain expertise about a robot's environment, sensor suite, and interactions between the two. Bayesian planning algorithms aim to navigate the exploration-exploitation tradeoff with respect to this distribution of environments that the robot anticipates seeing at test time.

We develop efficient Bayesian search algorithms motivated by *regret minimization*. This objective captures the urgency of a planner's sequential decision-making process by comparing with the optimal decision-maker at each iteration. The cumulative difference, or regret, penalizes suboptimality at each iteration as well as the time expended to reduce that suboptimality. We demonstrate that algorithms based on posterior sampling are effective for Bayesian anytime lazy motion planning and Bayesian dynamic motion planning.

We propose a *variational inference* algorithm for roadmap optimization, which aims to match the distribution of roadmap samples to the target distribution of collision-free states. We demonstrate that optimized sparse roadmaps concisely approximate the uncertain environment and can be searched more efficiently than conventional uniform or low-discrepancy dense roadmaps.

Acknowledgments

I am immensely thankful to have had so much support on the winding road to this dissertation.

Thank you to Sidd Srinivasa for being such an encouraging mentor and tireless advocate. I especially appreciate his enthusiastic support for even my research-adjacent passions; those hours spent teaching and volunteering are among my most cherished experiences from graduate school. I'm glad to have him in my corner.

Sidd has an extraordinary ability to crystallize the essence of a problem so that others can see as clearly as he does. I've learned so much from our time polishing and refining those ideas and implementations, whether it was a clever algorithm or a bare-bones website. Thank you for shepherding this whole journey, and for coming with me to explore the other dissertations that might have been.

I thank all the members of my doctoral supervisory committee for their helpful advice: Byron Boots, Sanjiban Choudhury, Wen Sun, Aaron Marburg, and Sam Burden. Thank you to Brian Coltin for his refreshing perspective during and after the NASA Space Technology Research Fellowship that funded this work. Thank you to Jeff Mahler, Michael Laskey, and Florian Pokorny for introducing me to robotics research.

I am grateful to the people of the Personal Robotics Lab, past and present. Kindness and brilliance have been a reliable constant, even as the people, robots, and buildings have changed. Thank you to Clint, Henny, Jimmy, Laura, Mike, Oren, Rachel, Shervin, Shushman, Stefanos, and Vinitha for welcoming me into the lab at CMU. Thank you to my labmates at UW: Aditya, Amal, Christoforos, Daqing, Ethan, Gilwoo, Hanjun, Jeongseok, Kay, Matt B., Patrick, Rosario, Sanjiban, Schmittle, Sherdil, Tapo, Willie, and Youngsun. Thanks to HERB for the fun demos and IMDb credit. Thank you to Lisa and Anna for all their help, and thanks to Sidd for bringing us together.

Sanjiban and Gilwoo have been great friends, mentors, and collaborators. Our time working together was too brief, but incredibly enjoyable and fruitful. It's not a coincidence that the backbone of this dissertation is constructed from our ideas. I also owe a special

thank you to Rosario and Aditya for being wonderful housemates. I couldn't have asked for more lively entertainment and reflective conversations while in the comfort of my own home.

I was lucky to find so many amazing friends and co-conspirators throughout this journey. I am especially thankful for Naveena, Nicasia, Chris, Willie, Kentrell, Miranda, Peter W., Philip, Erin, Matt J., and Alex O. Our camaraderie sustained me even more than the countless delicious meals that we cooked and shared. Thank you also to my CMU family, especially Lauren, Tess, Catherine P., Eugene, James, and Brian O. Thank you to Nick W., Mohit, Kay, Zoey, Naveena, Chris, and Willie for the hours spent bringing a new and improved robotics course to life at the Allen School. Thank you to all the PARS/PAMS organizers and mentors for their service, particularly Miranda and Les. I'm incredibly proud of the strides we've made. Thank you also to Elise and the CSE graduate advising team for patiently answering all my questions.

Thank you to Naveena for all the adventures (big and small) and for the cheer that she brings into every room. I deeply admire how she persistently embodies her values; I am a more thoughtful person because of her friendship. Thank you to Nicasia for her generous spirit. The numerous noodles, pizzas, stories, and reactions that we've shared always brought a smile to my face. Thank you to Chris for the honesty and vulnerability that characterizes our friendship. I am in awe of their continued strength and tenacity. I'm also grateful for their impeccable taste in entertainment. Thank you to Peter W., Nicasia, Nathan, Amanda, Chris, Willie, Erin, and Matt J. for housing me during the month before my defense. I don't know how crossing the finish line could have been more difficult, but I guarantee that it would have been if not for their generosity. There have been undeniable challenges during this journey, but I've had so many moments of joy and delight with my friends. Thank you.

Finally, none of this would have been possible without the infinite love and care of my family. I owe so much to the village that raised me—my parents, grandparents, aunts, uncles, cousins, and sister. Thank you to my parents for the many phone calls, boundless curiosity about my research (and my life beyond it), and patience while I took the most indirect route through graduate school.

Contents

List of Figures

List of Tables

1
Introduction

As robots are introduced to a wider variety of real-world domains—factories, roads, and homes—they will be tasked with operating with incomplete knowledge of the environment. Sensors and perception algorithms may imprecisely detect furniture or other obstacles that a robot shares space with. Forecasting models may inaccurately predict the decisions of other actors and the consequences of a robot's actions. Despite these varied uncertainties, we must be able to depend on robots to remain safe and operational.

This dissertation considers the problem of *motion planning with uncertainty*, where a robot navigates to a goal without knowing the environment's exact obstacle geometry. Real-world environments are cluttered but structured—so-called "obstacles" were left there by people or nature—and a robot can use this physical understanding of the world to filter and extrapolate beyond its own history of observations. This structure can be formalized as a Bayesian posterior distribution over latent environment parameters.

Planning with uncertainty is fundamentally about balancing exploration with exploitation: should a robot act to gain information and reduce uncertainty, or should it simply brave the unknown? Roboticists frequently leverage their domain expertise to define good algorithmic abstractions (e.g., the sense-plan-act paradigm) and "carve nature at its joints" to make robots more robust to uncertainty. The environment itself may be rigidly controlled to simplify the demands placed upon an autonomous system, and improved perception algorithms or extra sensors may be able to reduce uncertainty to the point that it can effectively be ignored. Bayes-optimal formulations of planning under uncertainty aim to maximize the Bayesian value function, but are not a silver bullet for improved performance compared to well-designed and carefully-engineered systems.[1] Unfortunately, these algorithms are often computationally expensive to deploy. This is exacerbated by the limited computation time available for replanning when robots are in motion.

[1] In practice, theoretically-sound algorithms are often bolstered by pragmatic information-based heuristics or shaped reward functions.

1.1 Guiding Principles

How can uncertainty be efficiently incorporated into decision-making, without ballooning planning times on computationally-constrained systems and paralyzing robots into indecision? How can algorithms leverage the *structure* of uncertainty to efficiently plan high-quality paths? This dissertation considers these questions across the sampling-based motion planning framework.

Our first key insight is to view robot motion planning through an online learning lens. This theoretical formalism—borrowed from machine learning—describes the interleaved process of making decisions and receiving reward feedback from the environment [38]. In particular, the objective of *regret minimization* captures the urgency of a planner's sequential decision-making process by comparing with the optimal decision-maker at each iteration. The cumulative difference is the algorithm's regret, which penalizes suboptimality at each iteration as well as the time an algorithm takes to reduce that suboptimality.[2] We demonstrate that search algorithms motivated by this objective are effective on two classes of Bayesian motion planning problems, anytime lazy motion planning and dynamic motion planning.

[2] A Bayes-optimal algorithm that requires significant computation time before yielding the optimal solution may incur more regret than an algorithm that quickly yields a near-optimal solution and takes longer to compute the optimal solution.

The core abstraction of sampling-based motion planning is the roadmap. Our second key insight is to leverage uncertainty to optimize the placement of roadmap samples: the distribution of roadmap samples should match the distribution of collision-free states. We demonstrate that optimized sparse roadmaps concisely approximate the uncertain environment and can be searched more efficiently than conventional uniform or low-discrepancy dense roadmaps. In this way, we integrate uncertainty throughout the sampling-based motion planning framework: an efficient roadmap post-processing step can help offset the increased computational demands of planning with uncertainty.

An underlying theme is that environment uncertainty—formalized by the posterior distribution—is structured and informative. This is a critical detail: roboticists must be able to express their domain expertise about a robot's environment, sensor suite, and interactions between the two. Bayesian algorithms aim to navigate the exploration-exploitation tradeoff with respect to the posterior, which characterizes the distribution of environments that the robot anticipates seeing at test time. Because the full space of all possible environments is too large to explore completely, a well-designed posterior distribution focuses the exploratory budget on environments that are most likely to be seen at test time and are thus important to solve efficiently.

1.2 Contributions

This dissertation makes the following contributions:

- A novel formulation of motion planning with uncertainty as Bayesian reinforcement learning (Chapter 2).
- The *Posterior Sampling for Motion Planning* algorithm, which establishes the first regret bounds for Bayesian anytime lazy motion planning (Chapter 3, Hou et al. [41]).
- The *Dynamic Replanning with Posterior Sampling* algorithm, which efficiently plans high-quality solutions to Bayesian dynamic motion planning problems (Chapter 4, Hou and Srinivasa [40]).
- The *Stein Variational Probabilistic Roadmap* algorithm, which leverages efficient particle-based variational inference to construct high-quality sparse roadmaps (Chapter 5, Lambert et al. [61]).
- Thorough experimental evaluations of the proposed algorithms on a range of simulated Bayesian motion planning problems.

2
Motion Planning with Uncertainty

This chapter describes the proposed framework for motion planning under uncertainty. Sampling-based motion planning is briefly reviewed in Section 2.1. Section 2.2 formalizes our definition of uncertainty by characterizing the Bayesian motion planning problem. In Section 2.3, we discuss approaches for representing and querying posterior distributions. The remaining chapters of this dissertation share the notation introduced in these sections. We also review related work for the Bayesian motion planning domains discussed in those chapters.

2.1 *Sampling-Based Motion Planning via Roadmaps*

Let $\mathcal{X}$ be the configuration space, which is composed of free space $\mathcal{X}_{\text{free}}$ and obstacle space $\mathcal{X}_{\text{obs}} = \mathcal{X} \setminus \mathcal{X}_{\text{free}}$. Given start and goal configurations $x_{\text{s}}, x_{\text{g}} \in \mathcal{X}_{\text{free}}$, a geometric path between them $\xi \colon [0,1] \to \mathcal{X}$ is a continuous function such that $\xi(0) = x_{\text{s}}$ and $\xi(1) = x_{\text{g}}$.[1] Let $\Xi(x_{\text{s}}, x_{\text{g}})$ be the set of all such paths. For a cost functional $w \colon \xi \to \mathbb{R}_{\geq 0}$, the optimal motion planning problem is:

$$\xi^* = \underset{\xi \in \Xi(x_{\text{s}}, x_{\text{g}})}{\arg\min}\ w(\xi) \text{ subject to } \xi(t) \in \mathcal{X}_{\text{free}},\ \forall t \in [0,1]. \tag{2.1}$$

[1] Unlike time-parameterized trajectories, geometric paths are arbitrarily parameterized between 0 and 1 to indicate that cost functionals can only evaluate the sequence of states (e.g., length).

A resulting optimal path is collision-free (or *feasible*) and minimizes w over the set of paths $\Xi(x_{\text{s}}, x_{\text{g}})$.

The computational intractibility of optimal motion planning in high dimensions has driven the development of sampling-based methods [89]. These approaches construct a discrete graph approximation $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to the continuous free space $\mathcal{X}_{\text{free}}$ by sampling configurations (vertices) and connecting them with potential motions (edges). Edge weights $w(e)$ are defined by evaluating the cost functional for the subpath connecting the incident vertices; an edge is collision-free if all configurations along the edge are collision-free. To solve a particular motion planning query $(x_{\text{s}}, x_{\text{g}})$, the two configu-

rations are inserted as vertices (v_s, v_g) into $\mathcal{G}$. Then, the problem of optimal motion planning on a roadmap $\mathcal{G}$ is:

$$\xi^* = \underset{\xi \in \Xi_{\mathcal{G}}(v_s, v_g)}{\arg\min} \sum_{e \in \xi} w(e) \text{ subject to } \xi(t) \in \mathcal{X}_{\text{free}}, \ \forall t \in [0,1]. \tag{2.2}$$

where $\Xi_{\mathcal{G}}(v_s, v_g)$ describes the set of paths between v_s and v_g.[2]

Two popular approaches for sampling-based motion planning are the Probabilistic Roadmap (PRM) and Rapidly-exploring Random Tree (RRT) algorithms [54; 58; 53]. The original RRT algorithm is designed to support single queries; the discrete tree approximation is constructed anew for each pair (x_s, x_g). On the other hand, the PRM algorithm is designed for multiple queries in the same static environment. The expense of constructing the discrete roadmap approximation can be amortized across those queries. In other words, while PRM-style algorithms can reuse the roadmap approximation to $\mathcal{X}_{\text{free}}$, RRT-style algorithms typically rediscover the topology of $\mathcal{X}_{\text{free}}$ in response to each query. Both approaches can be effective in practice, but we focus on multi-query roadmap-based methods to facilitate planning time reductions via precomputation.

In many practical robotics applications, even finding the optimal collision-free path in the discrete approximation $\mathcal{G}$ may be challenging due to limited computation time or physical sensory limitations. While achieving low cost is still desirable in these settings, collision avoidance while navigating to the goal is the primary concern. We consider two flavors of this feasible motion planning problem in this dissertation. In anytime motion planning, the objective is to quickly find feasible paths and continue to optimize them as computation time permits [65]. In dynamic motion planning, discovering the optimal collision-free path requires exploration and replanning due to the uncertainty of the robot's environment [102]. Thus, the objective is simply to reach the goal while avoiding collisions. Both flavors can be framed as problems where there is uncertainty about the edge weight function: depending on whether the edge is in collision, it may either have the known value from the cost functional evaluated on the edge $w(e)$ or an infinite cost.

[2] Note there is a difference between *optimal* and *optimized* paths [62]: a minimum cost path on $\mathcal{G}$ is the result of an optimization procedure (i.e., optimized), but it is not a true optimal path through the configuration space. If vertices are generated with a uniform random or low-discrepancy sampling procedure, the resulting graph admits a *resolution-optimal* path. Incremental densification procedures sample more vertices to improve the effective roadmap resolution and asymptotically approach an optimal path.

This dissertation primarily considers planning on a fixed roadmap. Thus, optimal paths are considered only according to the discretization $\mathcal{G}$.

2.2 *Bayesian Motion Planning*

In the previous section, we considered motion planning for a robot in a known environment described by the configuration space $\mathcal{X}$. We will now extend our notation to address environment uncertainty. Let ϕ be latent parameters that fully characterize the robot's unknown environment.[3] For ease of exposition, we will interchangeably refer to ϕ as the robot's unknown environment. When planning

[3] This is a flexible parameterization that can be chosen by the practitioner to ease the computational burden of a posterior update. Although ϕ consists of the full environment occupancy grid in the most general case, it can also be simplified to a set of obstacle geometries and positions.

on roadmaps, an edge's collision status is a deterministic function of the environment; thus, an equivalent parameterization of ϕ in the roadmap setting is a vector of edge collision statuses. For compactness, we will also denote whether edge e is collision-free in environment ϕ with $\phi(e)$.

To view uncertainty through a Bayesian lens, we assume the existence of a prior $P(\phi)$ that describes the distribution of planning environments the robot will be tasked with. This may be obtained from past experience or derived from other domain-specific knowledge. Let ψ_t be the history of observations about the environment after t interactions.[4] Given this history, a Bayesian motion planning algorithm can compute a posterior $P(\phi|\psi_t)$ to inform its next interaction. At each iteration, the posterior distribution quantifies the algorithm's current uncertainty about the environment.

[4] In general, the history ψ_t consists of the raw sequence of sensor observations $\{z_1, \cdots, z_t\}$. For lazy motion planning problems where observations are the outcomes of edge evaluations, this may be simplified to $\{w(e_1), \cdots, w(e_t)\}$.

The Bayesian approach proposed in this dissertation aims to shift the design and analysis of motion planning algorithms toward expected-case rather than worst-case performance. Improving worst-case performance is difficult due to the inherent intractibility of the general motion planning problem.

2.2.1 *Online Learning*

In the general online learning framework, a planner iteratively makes decisions about how to interact with its environment. The environment's reward function is unknown; the reward associated with a decision is only revealed after the agent has committed to that decision. To judge how quickly an algorithm learns, we consider the multi-armed bandit setting [7]: in each round of learning, an agent pulls an arm (decision), receives a reward, and accumulates regret with respect to the optimal arm (decision) in hindsight. This theoretical framework has been used to analyze convex optimization algorithms [38] as well as model-predictive control algorithms [111].

In Bayesian motion planning problems, a decision is the proposed path ξ_k and the reward $\mathcal{R}(\xi;\phi)$ depends on the environment ϕ. A learning algorithm must infer the reward function by repeatedly interacting with ϕ; the regret objective for that process is

$$\textsc{Regret}(m;\phi) = \sum_{k=1}^{m} \Delta_k, \text{where } \Delta_k = \mathcal{R}(\xi^*;\phi) - \mathcal{R}(\xi_k;\phi).$$

This penalizes an algorithm's suboptimal decision at each iteration.

To achieve low expected regret, an online learning algorithm must balance exploration with exploitation. In the multi-armed bandit setting, the posterior sampling algorithm (or Thompson sampling) [106] navigates this tradeoff with a stochastic policy that pulls arms with

the posterior probability that they are optimal. Upper Confidence Bound (UCB) algorithms rely on optimism to find balance: the arm with the highest confidence interval upper bound is pulled at each iteration [3]. These strategies translate to the repeated episodic reinforcement learning setting as well, as demonstrated by algorithms like PSRL [81] and UCRL2 [50].

2.2.2 *Connections to Bayesian Reinforcement Learning*

First, we define an equivalent deterministic Markov decision process for each motion planning problem $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, s_1 \rangle$. A state $s \in \mathcal{S}$ corresponds to a vertex $v \in \mathcal{V}$, actions $\mathcal{A}(s)$ correspond to the set of adjacent edges, and transition function $s' = \mathcal{T}(s, a)$ is the adjacency matrix of the graph. The reward function $\mathcal{R}(s, a)$ is 0 if $s = v_g$, else it is $-w(e)$, where e is the edge associated with a. The initial state s_1 is v_s. A solution to the MDP is a partial policy $\mu \colon \mathcal{S} \to \mathcal{A}$ corresponding to a path ξ. The policy's value is $V_\mu(s_1) = \sum_{j=1}^{\tau} \mathcal{R}(s_j, \mu(s_j))$ is the negative path length $-w(\xi)$.

An unknown environment ϕ translates to an unknown reward function $\mathcal{R}$. In this corresponding Bayesian reinforcement learning problem, a learning algorithm must infer the reward function by repeatedly interacting with $\mathcal{M}$. A prior over worlds $P(\phi)$ maps to a prior over rewards $P(\mathcal{R})$. We can then define an equivalent partially observable Markov decision process (POMDP) for each Bayesian motion planning problem, where the belief $b(\phi)$ is defined as the current posterior distribution $P(\phi|\psi_t)$. Solving for the optimal policy tree is computationally intractable: a policy tree of depth D for the general Bayesian motion planning problem contains $\mathcal{O}(2^D|\mathcal{V}|^D)$ nodes. As an alternative to the POMDP formulation, we can also consider each Bayesian motion planning problem as a Bayes-Adaptive Markov decision process (BAMDP) [30]. In the BAMDP formulation, the transition and reward functions are parameterized by the latent environment ϕ. This structure in the posterior distribution can be exploited to plan more efficiently.

Equivalence with Bayesian Anytime Motion Planning As a planning algorithm evaluates edges $e_1, \cdots, e_N$, it uncovers a series of progressively shorter paths $\xi_1, \cdots, \xi_N$. The objective of anytime planning is to minimize the cumulative length of paths, i.e., $\sum_{i=1}^{N} w(\xi_i)$. The objective of a *Bayesian* anytime planning algorithm is to minimize the *expected* cumulative length of paths computed given the prior over the worlds $P(\phi)$. This problem is equivalent to a repeated episodic Bayesian reinforcement learning problem, where a single uncertain MDP is explored across multiple episodes [81].

Let the MDP horizon τ correspond to the maximum number of edges in a path. In each episode $i = 1, \ldots, m$, the agent executes policies $\mu_1, \cdots, \mu_m$, updates history ψ_i, and tracks the best discovered policy $\widehat{\mu}_i = \arg\max_{j=1,\ldots,i} V_{\mu_j}(s_1)$. We define an algorithm's regret to be the cumulative difference between the value of the optimal policy and the value of the best discovered policy after each episode.

$$\textsc{Regret}(m) = \sum_{k=1}^{m} \Delta_k = \sum_{k=1}^{m} \left[V_{\mu^*}(s_1) - V_{\widehat{\mu}_k}(s_1) \right] \tag{2.3}$$

The objective of the Bayesian RL problem in Osband et al. [81] is to minimize the expected regret $\mathbb{E}_{P(\mathcal{R})}\left[\textsc{Regret}(m)\right]$, also known as the Bayesian regret. Note that this is a constant offset from the objective defined for Bayesian anytime planning:

$$\mathbb{E}_{P(\mathcal{R})}\left[\textsc{Regret}(m)\right] = \mathbb{E}_{P(\phi)}\left[\sum_{k=1}^{m} w(\xi_k) - w(\xi^*) \right]. \tag{2.4}$$

The MDP we have defined allows us to establish equivalences between RL and other lazy motion planning formulations proposed in previous work. The lazy *shortest path problem* [17], where the shortest feasible path must be found while eliminating all shorter paths, is equivalent to the PAC-MDP [103] problem of optimally exploring an MDP until an optimal policy is found. Similarly, the Bayesian version of this problem is equivalent to PAC-BAMDP [30]. The *feasible path problem* [13] is equivalent to Bayes-optimally exploring the MDP until a valid policy is found.

Equivalence with Bayesian Dynamic Motion Planning In dynamic motion planning, a robot must replan as it moves and observes previously unknown obstacles [102]. At each iteration, it follows a proposed path until traversing an edge would result in a collision. After this edge is detected, it replans a new path proposal. The objective of dynamic motion planning is to minimize the total distance traveled $w(\xi)$ before reaching the goal. As before, the objective of Bayesian dynamic motion planning is to minimize the expected total distance traveled by the robot $\mathbb{E}_{P(\phi)}\left[w(\xi)\right]$. This problem is equivalent to non-episodic Bayesian reinforcement learning: the robot must be able to recover from its own mistakes, rather than relying on the the environment to reset between episodes. Thus, the episode only terminates when the robot reaches the goal, regardless of how many detours it takes.

2.3 *Interfaces for Posterior Distributions*

This Bayesian formalism is agnostic to different sources of environment uncertainty: the posterior distribution is a statement about the

relationship between observations and latent environment parameters. In the case of lazy motion planning algorithms, which explicitly reason about when and where to perform the expensive operation of collision-checking, uncertainty is a consequence of *computational limitations* [37; 17; 33]. Even with access to the exact environment geometry, this computational bottleneck renders it intractable to completely eliminate uncertainty before beginning to plan. *Physical sensory limitations* may also be responsible for environment uncertainty, as distant regions may be inadequately captured by a robot's onboard sensor suite. This physical limitation makes it impossible for a robot to plan with complete certainty; a region that was previously thought to be collision-free may be blocked upon closer inspection.

Planning algorithms assume different interfaces to the posterior distribution. Some algorithms require the marginal posterior distribution $P(\phi(e) = 1|\psi_t) \propto \int_\phi P(\phi(e) = 1|\phi)P(\phi|\psi_t)$, which must be normalized to extract a posterior collision probability. This probability has a straightforward physical interpretation, but computing the partition function to normalize this distribution is intractable for many posteriors of interest. The search algorithms proposed in this dissertation require only that the posterior distribution is sampleable: $\phi \sim P(\phi|\psi_t)$. Posterior sampling ensures that only statistically plausible environments are sampled, while algorithms that consider marginal collision probabilities effectively take a weighted average across all plausible environments.[5]

[5] Consider I-5 during the summer: each day, one lane is randomly selected to be open while the remaining three lanes are closed for construction. All lanes have a marginal probability of closure of 0.75, demonstrating that they are each quite likely to be closed on any given day. However, this does not reflect that one lane will always be open.

The roadmap optimization algorithm we have proposed also avoids computing this partition function. Because only the gradient is necessary for updating the particles, it computes the gradient with respect to the negative log likelihood. The normalization term becomes an additive constant in the log likelihood, resulting in zero contribution to the gradient. Thus, the posterior distribution or log likelihood function only needs to be differentiable.

The algorithms presented in this dissertation require the posterior to be updated at the end of each iteration. However, optimistic algorithms do not require a full posterior update as long as newly-discovered edge blockages are accurately reflected. Updating the posterior distribution to incorporate new observations can be an expensive operation, especially for complex posterior distributions. There is a tradeoff between the computation time required to maintain the posterior and the information that can be extracted from the posterior. This is an important design decision for roboticists to consider: is planning with uncertainty required for their domain?

2.4 *Priors in Lazy Search*

Planning with expensive collision-checking is a well-studied problem in motion planning. Lazy search approaches address this by only checking edges that lie along the estimated shortest path [5; 37; 17] or the shortest subpath [16; 71; 72]. For real-world robotics problems, leveraging priors on edge collisions can produce significant speed-ups. FuzzyPRM evaluates paths that are most likely to be feasible [77]. GLS uses priors to quickly invalidate subpaths until the shortest path is found [72]. STROLL learns an edge evaluation policy for LazySP [4]. Choudhury et al. [14] formalizes Bayesian feasible path planning and shows that it is equivalent to the Bayesian active learning problem of decision region determiniation; a combination of offline DIRECT [11] with online BISECT [13] achieves state-of-the-art performance. However, these approaches do not aim for anytime performance.

Several methods estimate collision probabilities to infer edge collision statuses while planning. One approach is to predict validity of unevaluated edges given the outcomes of evaluated edges [23]. Other approaches try to model the configuration space belief given observed collisions and guide search with that belief [8; 82; 46; 60]. However, these approaches do not formalize these collision probability estimates as a Bayesian posterior.

2.5 *Anytime Planning*

For many real-time planning applications, an algorithm must be able to deal with an unknown planning time budget. Algorithms can guarantee asymptotic optimality with continued sampling even after discovering a feasible path [53; 1], but they make no promises on convergence rate and are often slow in practice. Incremental densification techniques restrict new samples to a region that can only improve the current solution, offering better sample efficiency [25; 28; 29]. However, these methods cannot provably exploit collision probabilities over the configuration space.

Another way of viewing anytime planning is through the lens of heuristic search on large graphs. Weighted A* search with an inflated heuristic finds feasible paths quickly, although the solution may be suboptimal [36]. Anytime variants of A* efficiently run a succession of weighted A* searches with decreasing inflation [65; 109]. However, heuristics may not always indicate existence of feasible paths. POMP uses priors on edge validity to explicitly tradeoff path likelihood and path length [12]. AEE* uses Bernoulli priors on edges to generate a set of plausible shortest paths, which is then evaluated in an anytime

fashion [76]. However, they do not offer theoretical guarantees for arbitrary priors.

2.6 *Dynamic Replanning with Uncertainty*

Variants of D* and D* Lite have been widely deployed in uncertain environments [102; 56; 26]. As edge costs change (e.g., when new obstacles are perceived), the search tree is repaired to dramatically reduce replanning time while remaining functionally equivalent to replanning from scratch with A*. D*-based algorithms determinize optimistically in the face of uncertainty; space is assumed to be collision-free until the robot perceives obstacles. In uncluttered scenarios where this assumption is generally warranted, exploiting the optimistic path is an excellent heuristic that sidesteps most of the computational expense of modeling uncertainty (e.g., updating the Bayesian posterior). Unlike many challenging POMDP domains that may demand that the robot divert itself from its goal to reduce uncertainty, simply following a path through uncertain regions provides valuable information in Bayesian motion planning problems.

The Canadian Traveler's Problem describes a scenario where edge blockages are discovered only upon arriving at an incident node. Planning a path that achieves a fixed suboptimality ratio to the shortest path is intractable [83]. Similarly, in dynamic motion planning problems, uncertainty is a consequence of sensor limitations. Edge blockages can be perceived only when the robot is physically nearby. Though just the nearby blockages can be perceived, those measurements enable the planner to infer other plausible blockages via the posterior. Stochastic variants of the Canadian Traveler's Problem have been proposed, where edge costs are estimated independently [24; 15], modeled by a Gaussian Process [20], or modeled using a black-box Bayesian posterior [66]. The Blindfolded Traveler's Problem is a specific Bayesian dynamic motion planning problem, where the only source of information is contact feedback and correlations are described with an approximate posterior [96]. While these Bayesian problems remain theoretically intractable to solve optimally, approximate algorithms, such as Hedged Shortest Path under Determinization (HSPD), can achieve high-quality solutions that navigate the exploration-exploitation tradeoff [66].

As with POMDPs, Bayesian motion planning problems can be solved either offline or online. In the Reactive Planning Problem, a complete policy must be computed offline under uncertainty about which edges are blocked. In this problem, each edge also has an associated sensing cost to determine whether it is blocked. Although solving the Reactive Planning Problem is intractable, mutual infor-

mation policies that trade off exploration and exploitation produce effective approximate solutions to this problem [70]. The Learned Reactive Planning Problem extends this to a lifelong setting by allowing the policy to adapt to previously observed obstacles across multiple episodes of navigation [108]. Algorithms for both versions of the Reactive Planning Problem rely on a fixed set of possible edge subsets, which can be filtered as edge statuses are observed.

2.7 *Bayesian Reinforcement Learning*

Standard reinforcement learning approaches consider optimal exploration of an unknown MDP until an optimal policy is computed. In the absence of prior knowledge, PAC-MDP approaches result in exhaustive experimentation in every possible state [103]. In the general Bayesian reinforcement learning (BRL) problem, the reward and transition functions of a Markov decision process are uncertain [30]. BRL algorithms typically target the objective of Bayesian regret [35; 7], which aims to bound the cumulative difference in expected reward between the optimal and learned policies. This differs from the objective of Bayes-optimality, which satisfies the Bellman optimality equation for POMDPs. Several approximation strategies have been proposed to solve this intractable problem [57; 9; 63]. Risk-averse measures such as conditional value at risk (CVaR) have also been proposed [90; 91].

UCRL2 determinizes the BRL problem with optimism in the face of uncertainty by following the optimal policy for an optimistic MDP given its observations thus far [50]. PSRL determinizes the BRL problem with posterior sampling by following the optimal policy for a sampled plausible MDP [104; 81; 110]. In practice, PSRL seems to outperform optimistic algorithms; Osband and Van Roy hypothesize that current algorithms perform unnecessary exploration as a consequence of excessive optimism [80].

3
Bayesian Anytime Motion Planning

This chapter is adapted from Hou et al. [41].

In this chapter, we formalize the problem of anytime motion planning. Anytime algorithms should quickly find feasible paths and shorten them as time permits [65]. Existing algorithms typically make *asymptotic* guarantees [53] that they will eventually find the optimal path, but practitioners often prefer heuristic approaches to these theoretically-sound algorithms due to better performance with their limited computational budget. Thus, this analysis leaves several practical questions unanswered. Given a budget of computation time, how suboptimal will the resulting path be? How will increasing the computation budget improve the quality of the solution? Formalizing these questions helps us better understand important *anytime* properties, not just asymptotic properties. Ultimately, this will enable practitioners to make more informed choices about the algorithms they deploy.

3.1 Anytime Lazy Motion Planning

Consider a robot manipulator, endowed with complete information about its static environment as it plans the shortest path to a specified goal configuration. Even in this relatively simple scenario, the computational bottleneck of collision-checking[1] introduces an element of uncertainty. Lazy motion planning algorithms explicitly reason about when and where to perform that expensive operation [37]. Despite access to the exact environment geometry, this bottleneck renders it intractable to completely eliminate uncertainty before beginning to plan. Thus, the collision statuses of unevaluated edges is unknown during search. We represent this uncertainty with a posterior distribution of edge collisions, conditioned on the environment observations.

[1] Collision checking answers the geometric question of whether a robot placed in a particular configuration would intersect with its environment. Furthermore, to check whether a robot can move between two collision-free configurations, the most common strategy is to densely interpolate along that movement and check each individual configuration. Depending on the collision-checking resolution, this often becomes a major bottleneck for motion planning algorithms.

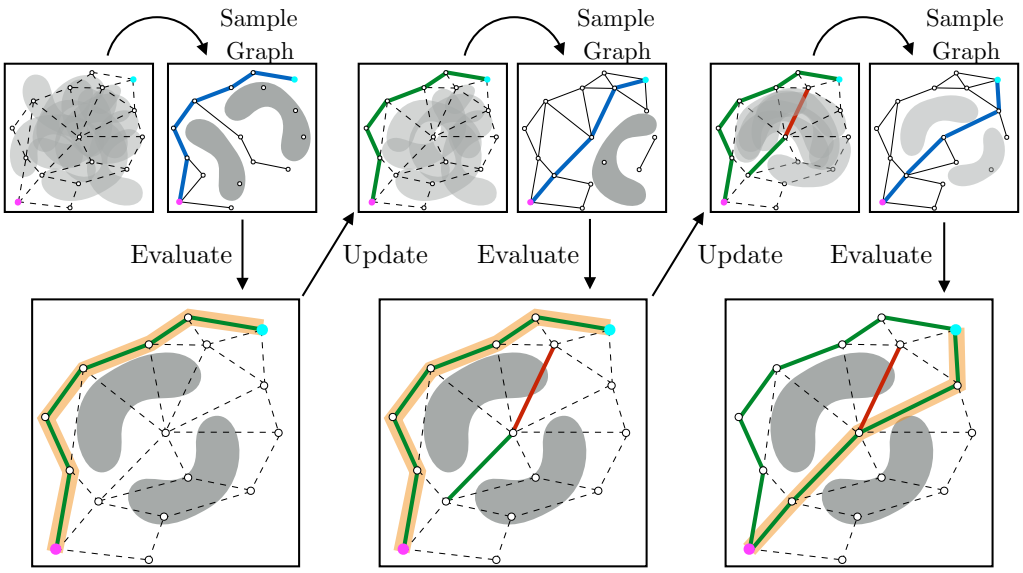

Figure 3.1: Posterior Sampling for Motion Planning [41] is a Bayesian anytime motion planning algorithm. The graph's edge collision statuses are initially unknown (dashed). In each iteration of PSMP, the posterior (top left) is sampled to produce a graph (top right). The sampled graph's shortest path (blue) is evaluated for collisions against the real world (bottom). Edges are either found to be in collision (red) or collision-free (green); those statuses are used to update the posterior. If all edges in the proposed path are collision-free, PSMP updates its current shortest path (yellow), which can be emitted at any time.

Optimism in the face of collision uncertainty minimizes the number of checks before finding the shortest path [17; 33]. However, this may take a prohibitively long time to compute with no other feasible paths discovered during this period. For many real-time applications, our goal is to quickly find feasible paths and shorten them as time permits—we refer to this as anytime lazy search [65]. The search must consider two factors: the length of a path and the likelihood of it being in collision. A desirable outcome, shown in Figure 3.1, is to initially evaluate longer paths that have lower probability of collision. Eventually, as uncertainty collapses, the search evaluates shorter and shorter paths. This strategy encapsulates a fundamental trade-off: an algorithm can either explore shorter paths to potentially improve future performance or exploit the most likely path to attain better immediate performance.

We formalize this problem within the framework of Bayesian reinforcement learning (BRL). To judge how quickly an algorithm learns, we consider the multi-armed bandit setting [7]: in each round of learning, an agent pulls an arm (evaluates a path), receives a loss (negative of path length), and accumulates regret with respect to the optimal arm. A low expected regret [35] corresponds to evaluating edges that not only lead to shorter paths, but also drive down uncertainty over time. Our key insight is:

Strong anytime search performance is equivalent to minimizing Bayesian regret.

However, the space of paths is combinatorially large, which makes many bandit algorithms that require explicit posteriors inapplicable. Fortunately, while explicitly computing this posterior is hard, sampling from it is quite easy. Posterior sampling offers strong guarantees on Bayesian regret [81]. Our proposed algorithm, Posterior Sampling for Motion Planning (PSMP), samples a graph from the posterior and only evaluates edges along the shortest path in that graph. It is both simple to implement and—given a posterior to sample from—free of tuning parameters. We show that PSMP achieves good theoretical anytime performance by bounding its Bayesian regret. Most importantly, we demonstrate that PSMP effectively leverages posteriors to outperform comparative baselines on a benchmark of 2D and 7D motion planning problems.

3.2 *Experienced Lazy Path Search*

We begin by presenting a framework for lazy search algorithms that uses priors on edge validities to minimize collision checking, thus unifying several previous works in this area [17; 12; 13; 4]. In Experienced Lazy Path Search, a *path proposer* lazily computes a path from the start to goal (without any edge evaluation) and a *path validator* chooses edges along the path to evaluate (Algorithm 3.1).[2] The two components take turns proposing and validating paths until the planning termination criteria is met (e.g., planning time budget has been expended). As soon as a path is validated, it is emitted as a feasible path; the last emitted path before the termination criteria was met is the best path discovered during planning.

[2] Note this unifying framework differs from the framework in Generalized Lazy Search (GLS) [72]. First, GLS looks at problems where planning time depends on *both* graph operations and edge evaluations. Hence, it argues for *interleaving* search with evaluation of *sub-paths*. Second, GLS exclusively considers the shortest path problem.

We fix the path validator to the FailFast rule [72] for all proposers (Algorithm 3.1, Lines 4–7). This rule tries to invalidate a proposed path as quickly as possible, formally stated as follows:

Theorem 3.2.1. *The* FailFast *validator repeatedly evaluates the edge with highest probability of collision, until one edge is found to be in collision or all edges are found to be collision-free. This is optimal for eliminating a single candidate path if prior $P(\phi)$ is independent Bernoulli.*[3] *For general priors, this is near-optimal with a factor of 4.*

[3] However, it ignores overlap among paths for simplicity, unlike Choudhury et al. [14].

Proof. This can be mapped to a Bayesian search problem where the goal is to sequentially search for an item (invalid edge) in a set of boxes (unevaluated edges) while minimizing cost of search. Bounds follow from Dor et al. [22]. □

Require: Graph $\mathcal{G}$, Prior $P(\phi)$, Proposer ComputePath$(\cdot)$
1: Initialize history $\psi \leftarrow \emptyset$, evaluated edges $\mathcal{E}_{\text{eval}} \leftarrow \emptyset$
2: **while** termination criteria not met **do**
3: Invoke proposer $\xi =$ ComputePath$(\mathcal{G}, P(\phi|\psi))$.
4: **while** path ξ is not invalid **and** ξ is unevaluated **do**
5: Evaluate unevaluated edge with highest posterior
6: collision probability $e^* = \arg\min_{e \in \xi \setminus \mathcal{E}_{\text{eval}}} P(\phi(e) = 1|\psi)$.
7: Add edge to evaluated set $\mathcal{E}_{\text{eval}} \leftarrow \mathcal{E}_{\text{eval}} \cup \{e^*\}$.
8: Update history ψ with outcomes.
9: **if** ξ is valid **then** emit ξ.

Algorithm 3.1: Experienced Lazy Path Search. This framework unifies several lazy search algorithms that use priors on edge validities to minimize collision checking.

By varying the proposer ComputePath, we can recover several lazy search algorithms from the literature that target different performance guarantees. All proposers listed in Table 3.1 view $\mathcal{G}$ with modified edge weights and propose the minimum weight path.

- LazySP [17] returns the optimistic shortest path by modifying all unevaluated edges in $\mathcal{G}$ to be feasible. It eliminates candidate paths in order of increasing length and terminates after a feasible path is found, yielding an OFU-like guarantee of finding the shortest path with minimal evaluations.
- MaxProb [13] returns the most likely feasible path by modifying the weights to be the negative log likelihood of validity. It sequentially evaluates the most probable path, terminating after a feasible path is found. Similar to Theorem 3.2.1, MaxProb Bayes-optimally proposes the fewest paths before finding a feasible path if prior $P(\phi)$ is independent Bernoulli and near Bayes-optimally (with a factor of 4) otherwise.
- Finally, POMP [12] balances edge weight with the likelihood of being collision-free. Increasing α between iterations of Algorithm 3.1 traces out the Pareto frontier of the two objectives, starting with the most probable path (equivalent to MaxProb) while guaranteeing asymptotic optimality. However, this anytime heuristic comes without guarantees on rate of improvement.

3.2.1 *Posterior Sampling for Motion Planning (PSMP)*

PSMP aims to provide strong guarantees on anytime behavior. It essentially borrows the idea of posterior sampling [81], or Thompson sampling [106], and applies it in the space of paths. PSMP samples a world ϕ from the posterior distribution $P(\phi|\psi)$ conditioned on the history ψ. It then computes the shortest path ξ^* on ϕ and proposes it for evaluation.

Algorithm	ComputePath($\mathcal{G}, P(\phi\|\psi)$)	**Performance Guarantee**
PSMP [41]	Sample a world $\phi \sim P(\phi\|\psi)$	Anytime (Bayesian regret)
LazySP [17]	Generate optimistic world ϕ	Shortest path (OFU)
MaxProb [13]	Set weights $-\log P(\phi(e) = 1\|\psi)$	Feasible path (Bayes-optimal)
POMP [12]	Set weights $\alpha w(e) - (1-\alpha)\log P(\phi(e) = 1\|\psi)$	Anytime (Pareto optimality)

Table 3.1: Different lazy search algorithms as instantiations of Experienced Lazy Path Search.

From an algorithmic perspective, PSMP is attractive because it requires solving only a single shortest path problem. By contrast, other Bayesian search methods like Monte Carlo Tree Search [32] or even heuristics like QMDP [68] require several calls to the search. PSMP also requires no tuning parameters. By sampling paths according to the posterior probability they are optimal, PSMP continues to sample plausible shortest paths. As PSMP gains more information, the posterior concentrates around the true world.

Following the analysis of posterior sampling for reinforcement learning [81] and multi-armed bandits [93], we will now establish Bayesian regret bounds for PSMP. The regret for PSMP grows sublinearly as $\widetilde{O}(\sqrt{\mathcal{SA}T})$ where T is the total number of timesteps, matching the lower bound from Jaksch et al. [50].[4]

[4] For this analysis, we assume edge weights are normalized between $[0, 1]$.

Theorem 3.2.2. *The expected regret is bounded as*

$$\mathbb{E}\left[\textsc{Regret}(T)\right] = O(\tau\sqrt{\mathcal{SA}T\log(\mathcal{SA}T)}) \tag{3.1}$$

Proof. We follow the analysis of Osband et al. [81], adapted for the special case of a deterministic MDP to obtain tighter regret bounds. $V_\mu^{\mathcal{M}}$ denotes the value of policy μ in the MDP $\mathcal{M}$. $\mathcal{M}_k$ refers to the sampled MDP at episode k and $\mathcal{M}^*$ is the underlying (unknown) MDP being repeatedly explored during all episodes.

We begin by noting that regret is measured with respect to the best discovered policy $\widehat{\mu}_k$ which is history-dependent, i.e., dependent on $(\mu_1, \mu_2, \ldots, \mu_k)$. Hence, we upper bound it with an alternative version of regret with respect to the executed policy μ_k.

$$\Delta_k = V_{\mu^*}^{\mathcal{M}^*}(s_1) - V_{\widehat{\mu}_k}^{\mathcal{M}^*}(s_1) \leq V_{\mu^*}^{\mathcal{M}^*}(s_1) - V_{\mu_k}^{\mathcal{M}^*}(s_1) \leq \bar{\Delta}_k \tag{3.2}$$

Posterior sampling leverages the fact that $\mathcal{M}^*$ and $\mathcal{M}_k$ are identically distributed. One hurdle in the analysis is that the optimal policy μ^* is not directly observed. Hence, we introduce yet another notion of regret which does not depend on μ^*.

$$\widetilde{\Delta}_k = V_{\mu_k}^{\mathcal{M}_k}(s_1) - V_{\mu_k}^{\mathcal{M}^*}(s_1) \tag{3.3}$$

which is the difference in expected value of the policy μ_k under the sampled MDP $\mathcal{M}_k$ and the true MDP $\mathcal{M}^*$ which is observed. We

apply Theorem 2 from Osband et al. [81] to show that the two regrets are equal in expectation

$$\mathbb{E}\left[\sum_{k=1}^{m} \bar{\Delta}_k\right] = \mathbb{E}\left[\sum_{k=1}^{m} \widetilde{\Delta}_k\right] \tag{3.4}$$

with high probability.

We will now bound $\mathbb{E}\left[\sum_{k=1}^{m} \widetilde{\Delta}_k\right]$. Unlike the analysis in Osband et al. [81] for the stochastic case, the deterministic regret is much easier to bound: it amounts to the difference in rewards observed in $\mathcal{M}^*$ versus $\mathcal{M}_k$. We will bound this by arguing that $\mathcal{M}^*$ concentrates around $\mathcal{M}_k$ using the notion of confidence sets as in Jaksch et al. [50]. Let t_k be the time at the beginning of the k^{th} episode. Let $\widehat{\mathcal{R}}(s,a)$ be the empirical average reward and $N_{t_k}(s,a)$ be the number of times (s,a) was queried. We define the confidence set for episode k as:

$$\mathbf{M}_k = \left\{\mathcal{M}\colon \left|\widehat{\mathcal{R}}(s,a) - \mathcal{R}^{\mathcal{M}}(s,a)\right| \leq \beta_k(s,a) \quad \forall (s,a)\right\} \tag{3.5}$$

where $\beta_k(s,a) = \sqrt{\frac{7\log(2\mathcal{S}\mathcal{A}mt_k)}{\max\{1,N_{t_k}(s,a)\}}}$ is chosen to ensure both $\mathcal{M}_k$ and $\mathcal{M}^*$ belong to $\mathbf{M}_k$ with high probability as specified in Jaksch et al. [50]. We bound regret as follows:

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i=1}^{m} \widetilde{\Delta}_k\right] \\
&\quad \leq \mathbb{E}\left[\sum_{k=1}^{m} \widetilde{\Delta}_k \mathbb{I}(\mathcal{M}_k, \mathcal{M}^* \in \mathbf{M}_k)\right] + 2\tau \sum_{k=1}^{m} P(\mathcal{M}^* \notin \mathbf{M}_k) \\
&\quad \leq \mathbb{E}\left[\sum_{k=1}^{m} \mathbb{E}\left[\widetilde{\Delta}_k | \mathcal{M}^*, \mathcal{M}_k\right] \mathbb{I}(\mathcal{M}_k, \mathcal{M}^* \in \mathbf{M}_k)\right] + 2\tau \\
&\quad \leq \mathbb{E}\left[\sum_{k=1}^{m} \sum_{i=1}^{\tau} \min\{\beta_k(s_{t_k+i}, a_{t_k+i}), 1\}\right] + 2\tau \\
&\quad \leq \min\{\tau \sum_{k=1}^{m} \sum_{i=1}^{\tau} \min\{\beta_k(s_{t_k+i}, a_{t_k+i}), 1\}, T\}
\end{aligned} \tag{3.6}$$

where the second inequality follows from the fact that Lemma 17 of Jaksch et al. [50] shows $P(\mathcal{M}^* \notin \mathbf{M}_k) \leq \frac{1}{m}$. The final inequality follows from the fact that worst case regret is bounded by T. We now bound

$$\min\{\tau \sum_{k=1}^{m} \sum_{i=1}^{\tau} \min\{\beta_k(s_{t_k+i}, a_{t_k+i}), 1\}, T\} \tag{3.7}$$

First note that

$$\sum_{k=1}^{m} \sum_{i=1}^{\tau} \beta_k(s,a) \leq \sum_{k=1}^{m} \sum_{i=1}^{\tau} \mathbb{I}(N_{t_k} \leq \tau) + \sum_{k=1}^{m} \sum_{i=1}^{\tau} \mathbb{I}(N_{t_k} > \tau)\beta_k(s,a) \tag{3.8}$$

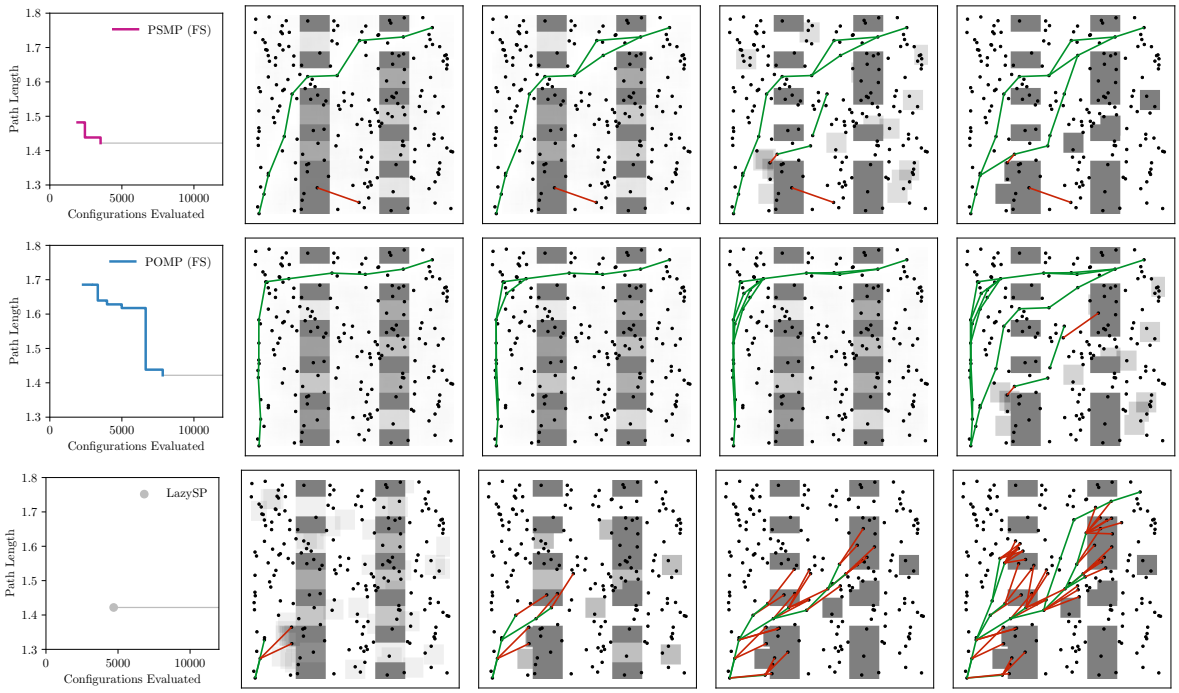

Figure 3.2: Anytime behavior of PSMP, POMP, and LazySP on an illustrative environment from the TwoWall dataset. (Left) Anytime performance curves. The gray line corresponds to the environment's shortest path. (Right) Snapshots of each algorithm's progress, along with the finite set posterior used by PSMP and POMP. Regions with higher probability of collision are colored with darker shades of gray. Evaluated edges are either found to be in collision (red) or collision-free (green).

The first term is shown to be bounded.

$$\sum_{k=1}^{m}\sum_{i=1}^{\tau}\mathbb{I}(N_{t_k}\leq\tau)\leq 2\tau\mathcal{SA} \tag{3.9}$$

The second term utilizes the following bound

$$\sum_{k=1}^{m}\sum_{i=1}^{\tau}\sqrt{\frac{\mathbb{I}(N_{t_k}>\tau)}{\max\{1,N_{t_k}(s,a)\}}}\leq\sqrt{2\mathcal{SA}T} \tag{3.10}$$

We can now bound

$$\begin{aligned}&\min\{\tau\sum_{k=1}^{m}\sum_{i=1}^{\tau}\min\{\beta_k(s_{t_k+i},a_{t_k+i}),1\},T\}\\&\leq\min\{2\tau^2\mathcal{SA}+\tau\sqrt{14\mathcal{SA}T\log(\mathcal{SA}T)},T\}\\&\leq\tau\sqrt{16\mathcal{SA}T\log(\mathcal{SA}T)}\end{aligned} \tag{3.11}$$

Hence, the Bayesian regret is bounded by $O(\tau\sqrt{\mathcal{SA}T\log(\mathcal{SA}T)})$. □

3.3 *Experiments*

We evaluate the anytime performance of PSMP on 2-DOF point robot navigation [13] and 7-DOF manipulator planning datasets. The 7-DOF manipulator dataset was generated by randomly perturbing objects from an initial cluttered environment [86]. For 2-DOF problems, we compare PSMP and POMP with two different posterior variants. For 7-DOF problems, we only consider the finite set posterior for both PSMP and POMP. In this domain, we additionally combine RRTConnect with path shortening, a commonly-used heuristic for refining an initial feasible path [58]. Because collision checking dominates planning time, we report the number of configurations checked by each algorithm. Section 3.3.4 contains further experimental details.

Dataset	Vertices	Edges
OneWall	100	923
TwoWall	200	2524
Forest	200	2524
MovingWall	150	1689
Maze	200	2524
Baffle	150	1689
Bugtrap	150	1689
Baxter (7D)	5002	137627

Table 3.2: Roadmap statistics for 2-DOF and 7-DOF motion planning datasets.

3.3.1 *Point Robot Navigation*

We visualize sample runs in a TwoWall environment in Figure 3.2. Note that since LazySP is not an anytime algorithm, it only produces one solution. Using the same finite set posterior as POMP, PSMP finds a shorter feasible path with fewer collision checks. Furthermore, it returns the shortest path faster than the uninformed LazySP baseline. POMP carefully attempts to avoid edges that may be in collision; as a result, refining the initial feasible solution can take a substantial amount of time.

Figure 3.3 summarizes the performance of these algorithms on seven different 2-DOF point robot navigation datasets. Each environment admits a different shortest path, so we overlay each anytime performance curve across the 200 environments in the test set. A desirable result is for the anytime performance curves to be pushed into the lower-left corner; this means that the algorithm quickly discovers short feasible paths and rapidly refines to the shortest paths. The anytime performance of the nearest neighbor-based (NN) and finite set (FS) posteriors have been separated for clarity. Finally, we also visualize the percentage of planning problems for which a fixed collision-checking budget is sufficient to discover a feasible path. A desirable result is for this curve to quickly reach 100%, i.e., that a small collision-checking budget is sufficient to discover (at least) a first feasible path.

First, we compare the performance of these planning algorithms when using the finite set posterior compared to the nearest neighbor-based posterior.[5] We find that PSMP-FS has a marked improvement over PSMP-NN across all datasets: it discovers initial feasible solutions faster than all other algorithms and quickly refines them. This boost demonstrates the importance of matching the posterior distri-

[5] The posterior distribution is an interface for practitioners to specify the intended distribution of planning problems. We use the FS posterior as an example of how domain knowledge about the robot's possible environments can be translated into a posterior distribution that helps the planning algorithm focus on plausible problems. In comparison, the NN posterior provides less information about the environment when conditioned on a particular observation history.

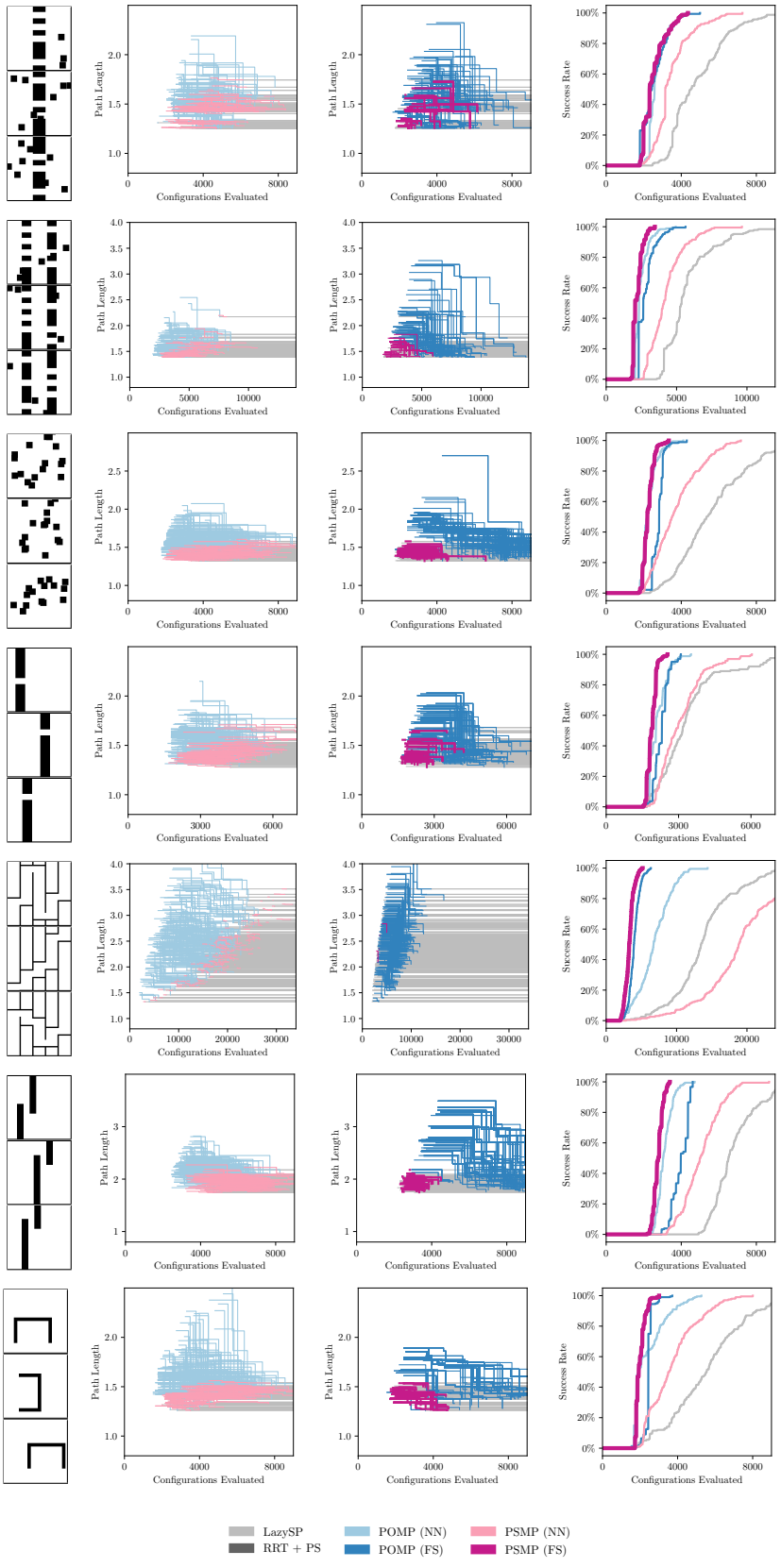

Figure 3.3: Anytime performance on point robot navigation datasets. (Left) Example problems from each dataset. (Center) Length of the best feasible path discovered by each anytime algorithm over time, using the nearest-neighbor posterior (column 2) and finite set posterior (column 3). (Right) Collision checking budget versus the percentage of planning problems where that budget is sufficient to discover a feasible path.

bution to the actual test-time distribution. The NN posterior is uncertain about environment regions that are distant from its collision observations. This causes the NN posterior to explore many possible samples for those uncertain regions, while the FS posterior can use the finite set of known environments to extrapolate more accurately. Curiously, POMP-FS does not consistently outperform POMP-NN; on the TwoWall and Baffle datasets, POMP-NN actually outperforms POMP-FS (Figure 3.3, Rows 2 and 7). We hypothesize that averaging the worlds in the feasible set that are consistent with the evaluation history—as POMP does—can lead to over-exploration of impossible scenarios and degraded performance.

PSMP-FS continues to outperform other algorithms on the Maze dataset (Figure 3.3, Row 5). With the FS posterior, the evaluation history quickly narrows the distribution to a single feasible world. Thus, PSMP-FS and POMP-FS enjoy similar performance. However, in many Maze environments, LazySP is able to discover the shortest path before PSMP-NN discovers its first feasible path. We believe that this is because an anytime objective implicitly assumes the existence of multiple feasible paths. Although posterior sampling explores options quickly, exploration may be unnecessary and undesirable if this assumption is violated. In such scenarios, a posterior that captures more global correlations in the environment may be needed for improved performance relative to an algorithm like LazySP.

3.3.2 *Baxter Manipulator Planning*

On the 7-DOF manipulator dataset, we have compared with heuristically shortening an initial feasible RRTConnect path (RRT+PS) rather than RRT* [53]. While RRT+PS finds a feasible path faster than LazySP finds the shortest path, it needs more collision checks and emits longer paths than PSMP or POMP with the FS posterior (Figure 3.4). We note that RRT+PS is a stronger baseline comparison than RRT*: although the latter is an anytime algorithm that guarantees asymptotic optimality, it empirically takes much longer to find initial solutions (Figure 3.5). Even with a significantly larger collision-checking budget, RRT* only finds a feasible path in 70% of the environments, although the initial feasible paths discovered are much shorter than those from RRTConnect.

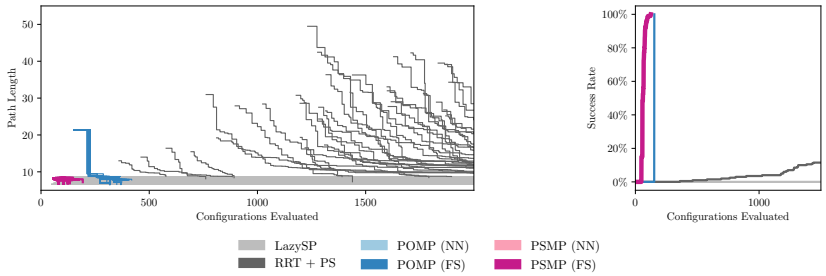

Figure 3.4: Anytime performance on 7-DOF robot manipulator datasets. (Left) Length of the best feasible path discovered by each anytime algorithm over time. (Right) Collision checking budget versus the percentage of planning problems where that budget is sufficient to discover a feasible path.

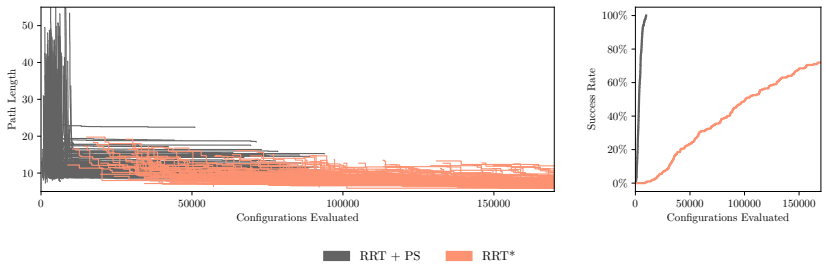

Figure 3.5: Compared to RRT+PS, RRT* requires many more collision checks before it can emit any feasible solutions. The x-axis of this plot is about 100 times larger than Figure 3.4.

3.3.3 *Estimating Edge Collision Posteriors*

In our experiments, we consider two possible approaches for estimating the posterior distribution $P(\phi|\psi)$. If there is no dataset of previous planning problems to learn from, the collision-checked configurations from ψ can inform a nearest neighbor-based posterior for the current problem [82]. We find that only considering the 1-nearest neighbor q_{near} produces the best collision estimates due to massive label imbalance in favor of collision-free points.

We differ from Pan et al. [82] by assuming a uniform Beta$(1,1)$ prior on configuration space collision probability. The status of the nearest neighbor counts as a partial success or failure with weight $\exp(-\eta\|q - q_{near}\|)$. The expected posterior probability that configuration q is free is then

$$\mathbb{E}\left[P(\phi(q) = 0|\psi)\right] = \frac{\exp(-\eta\|q - q_{near}\|)\mathbf{1}[\phi(q_{near}) = 0] + 1}{\exp(-\eta\|q - q_{near}\|) + 2}. \quad (3.12)$$

To estimate the posterior probability that an edge is collision-free, we take the minimum collision-free probability of discretized points along the edge.

Alternatively, if we know that worlds are uniformly drawn from a finite set of possible worlds, we can precompute the collision statuses for every edge in the graph against every world ϕ. Then, the posterior is simply uniform over the remaining set of worlds that are consistent with ψ.

This finite set posterior is one example of how planning algorithms may be able to leverage the structure existing in everyday environ-

ments. The configuration space nearest-neighbor posterior only assumes that nearby configurations will have similar labels, which is a more broadly applicable (but less informative) structure. This makes it well-suited for novel environments where the posterior does not have problem examples to learn from.

3.3.4 *Implementation Details*

Nearest Neighbor-Based (NN) Posterior For the 2-DOF point robot navigation environments, we use $\eta = 10^3$. To estimate the posterior probability that an edge is collision-free, we take the minimum collision-free probability of 5 discretized points along the edge.

Collision-Checking Each edge is collision-checked up to a fixed resolution via binary search. For the 2-DOF point robot navigation environments, both dimensions range from 0 to 1 and edges are checked at a resolution of 0.001. Edges are checked at a resolution of 0.2 for the 7-DOF manipulator planning environments.

POMP In each iteration, α controls the trade-off between collision probability and edge weight. The algorithm starts with $\alpha = 0$ and increases to $\alpha = 1$ as new feasible paths are discovered. We use the same step size of 0.1 evaluated in the original paper [12].

*RRT+PS and RRT** We used the OMPL implementations of RRTConnect and RRT*. On each of the 7-DOF environments, the algorithms search until they discover a path that is shorter than the shortest path in the graph (or until a maximum time limit has been exceeded). RRT+PS was evaluated with a 5 second timeout, while RRT* needed an increased limit of 30 seconds to return feasible solutions for all environments.

Path shortcutting is implemented with the OMPL default parameters. For both RRTConnect and RRT*, we set the range to infinity. Intermediate states were added to the tree for RRTConnect. For RRT*, we turned on lazy collision-checking and focused search (for pruning and informed sampling once a feasible path was discovered). We used the default RRT* rewiring factor of 1.1.

3.4 *Discussion*

Anytime algorithms should rapidly discover progressively shorter paths. We have formalized this intuitive objective as minimizing Bayesian regret. Sublinear Bayesian regret—which PSMP achieves—implies asymptotic optimality, while demanding good intermediate performance. We hope that drawing this connection between anytime motion planning and Bayesian reinforcement learning will open the door to further regret analysis of anytime algorithms.

Empirically, PSMP has strong performance and improves further when more knowledge is incorporated into the posterior structure. Comparative baselines were too conservative when faced with uncertainty and tried to avoid edges with any chance of collision. PSMP discovered and refined initial feasible solutions much more quickly with the finite set posterior. This chapter demonstrates that collision posteriors are valuable structures for embedding learned knowledge into planning algorithms, while highlighting the importance of how the posterior distribution is leveraged by planning algorithms.

Using data from previous planning environments to learn the underlying structure of collision posteriors (e.g., via unsupervised generative models) will enable PSMP to quickly solve new instances of those problems. We believe that this approach will be especially fruitful for combining motion planning with machine learning, to preserve guarantees of probabilistic completeness and asymptotic optimality while reducing expected planning time over the posterior distribution.

4
Bayesian Dynamic Motion Planning

This chapter is adapted from Hou and Srinivasa [40].

When navigating to a goal in an uncertain environment, a robot must simultaneously navigate the exploration-exploitation tradeoff: should it aim to gain information and reduce uncertainty, or should it simply brave the unknown? In the dynamic motion planning problem, a robot must replan as it moves and observes previously unknown obstacles [102]. However, the increased expense of reducing uncertainty requires that algorithms balance exploration with exploitation even more carefully. We now analyze several strategies from the literature and characterize how design choices within the algorithmic framework of *determinization* in the face of uncertainty impact path quality and planning efficiency.

4.1 Dynamic Motion Planning

In the dynamic motion planning problem, uncertainty is a consequence of physical sensory limitations. The robot is uncertain about areas that it has not yet observed and must physically move to reduce uncertainty.[1] However, it may also be able to leverage the posterior distribution to extrapolate and infer regions that may be blocked by obstacles.

[1] Anytime lazy motion planning can be viewed as dynamic motion planning where the robot can teleport. Then, traversal time incurs negligible cost and reducing uncertainty in any region of the environment becomes equally expensive. As a corollary, the path validator in anytime lazy motion planning can validate any edge that it desires for maximum efficiency; because dynamic motion planning requires physical traversal, the analogous validator is constrained to contiguous subpath segments and may even revalidate segments while backtracking.

We formalize this task as another Bayesian motion planning problem, where the robot's understanding of its uncertain environment is captured by a posterior distribution conditioned on its past observations. Thus, its behavior must adapt to its current position and posterior distribution over environments.

This general Bayesian dynamic motion planning problem can be described as a partially observable Markov decision process (POMDP). Solving for the optimal policy tree is computationally intractable: an offline policy tree of depth D for Bayesian motion planning contains $\mathcal{O}(2^D|\mathcal{V}|^D)$ nodes due to the "curse of history." Rather than preparing a response for all possible action-observation sequences, *online* POMDP methods interleave planning and execution

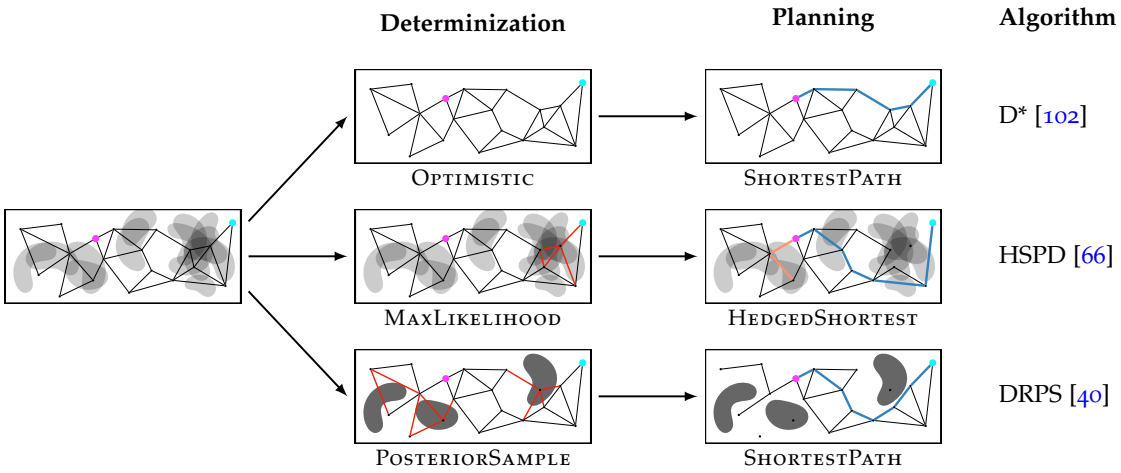

Figure 4.1: Determinization in the face of uncertainty is a popular framework for solving challenging Bayesian dynamic motion planning problems. The probabilistic problem (left) reflects uncertainty about obstacles (gray). In this framework, algorithms construct a deterministic estimate of the uncertain environment by removing edges from the graph (red) and assuming the remaining edges are collision-free. Because uncertainty was eliminated by *determinization*, *planning* becomes more tractable. To navigate the exploration-exploitation tradeoff, HSPD explicitly plans an exploration path (orange) and exploitation path (blue) and follows the shorter of the two. DRPS shifts the burden of exploration to determinization, which simplifies each iteration of planning to a single shortest path query and yields significant performance gains.

only for the current information state [92]. This approach is critical for scaling to large POMDPs [99; 101; 105].

POMDP algorithms are designed to address uncertainty across a variety of sources, including the robot's state and the transition and reward functions. However, factored POMDP models provide structure to uncertainty that can be leveraged to plan more efficiently [30; 79; 9]. In Bayesian dynamic motion planning, uncertainty originates only from the robot's ignorance about the environment; the transition and reward function are deterministic given the environment, and the robot's state is fully observable.[2]

We unify several online algorithms for dynamic motion planning under uncertainty within a framework of *determinization in the face of uncertainty* (Figure 4.1). These problems are often solved via a common pattern where planning on a simplified model is interleaved with feedback that improves the simplified model. The simplified model, or *determinization*, removes uncertainty to construct an approximation that can be solved more efficiently. For Bayesian motion planning problems, a determinization strategy assigns a collision status to every vertex and edge in the roadmap. Determinization is a successful and well-studied heuristic that takes advantage of structure in probabilistic planning problems [114; 115; 101].

We analyze popular D*-based optimistic approaches [102; 56] and uncertainty-aware Bayesian methods [85; 66] through this lens. Our key insight is:

> *The determinization strategy is critical for efficiently balancing exploration with exploitation.*

If the deterministic approximation preserves too many possible paths, planning must expend additional effort to avoid overexploration. Conversely, if determinization prunes too many exploration

[2] As with anytime lazy motion planning, this would be a simple graph search problem if the environment were fully observable.

Require: Graph $\mathcal{G}$, Prior $P(\phi)$
1: **while** goal not reached **do**
2: Determinize BDMP based on $P(\phi|\psi_t)$ ▷ Algorithm 4.2
3: Plan path $\hat{\xi}_{t+1}$ in determinized $\widehat{\mathcal{G}}$ ▷ Algorithm 4.3
4: Follow path until collision or end is reached
5: Update ψ_{t+1} with observations

Algorithm 4.1: Determinization for online Bayesian dynamic motion planning.

options, planning cannot retroactively correct for this imbalance. Choosing the right determinization has significant ramifications on planning efficiency and Bayesian regret.

We reinterpret posterior sampling [106] as a determinization strategy that effectively navigates this tradeoff. Posterior sampling achieves excellent empirical and theoretical performance in multi-armed bandits, reinforcement learning, model predictive control, and motion planning [81; 110; 41].

This chapter introduces Dynamic Replanning with Posterior Sampling (DRPS), an algorithm for Bayesian dynamic motion planning that shifts the burden of exploration to determinization rather than planning. This new strategy outperforms relevant baselines for a wide variety of 2D and 7D motion planning benchmark problems, with a lower or comparable total distance traveled and significant gains in computation time. These improvements are especially prominent on larger 7D planning problems, which suggests that relying on determinization for exploration is a promising direction for continued work.

4.2 *Efficient Probabilistic Planning via Determinization*

Algorithm 4.1 summarizes the framework of determinization in the face of uncertainty, which unifies several algorithms for online BDMP (Figure 4.1). Algorithm 4.2 characterizes how these algorithms choose to determinize. Given a posterior distribution that describes each edge's probability of collision, the determinization strategy approximates each edge as either blocked or collision-free. In the resulting roadmap, the statuses of each edge are known; Algorithm 4.3 describes algorithms for planning through the determinized roadmap.

4.2.1 *Balancing Exploration and Exploitation*

Dynamic A (D*)* D* plans the shortest path from the current vertex to the goal on an optimistic determinization of the roadmap. This simple approach sidesteps most of the computational expense of modeling uncertainty (e.g., updating the Bayesian posterior), and is

procedure OPTIMISTIC($\mathcal{G}, P(\phi|\psi_t)$)
  $\widehat{\mathcal{E}} = \{e \in \mathcal{E} \mid P(\phi(e) = 1|\psi_t) > 0\}$
  return $\mathcal{V}, \widehat{\mathcal{E}}$
procedure MAXLIKELIHOOD($\mathcal{G}, P(\phi|\psi_t)$)
  $\widehat{\mathcal{E}} = \{e \in \mathcal{E} \mid P(\phi(e) = 1|\psi_t) \geq 0.5\}$
  return $\mathcal{V}, \widehat{\mathcal{E}}$
procedure POSTERIORSAMPLE($\mathcal{G}, P(\phi|\psi_t)$)
  $\widehat{\phi} \sim P(\phi|\psi_t)$
  $\widehat{\mathcal{E}} = \widehat{\phi}(\mathcal{E})$
  return $\mathcal{V}, \widehat{\mathcal{E}}$

Algorithm 4.2: Determinization strategies.

procedure SHORTESTPATH($v, v_g, \widehat{\mathcal{G}}$)
  return A*$(v, v_g, \widehat{\mathcal{G}})$
procedure HEDGEDSHORTEST($v, v_g, \widehat{\mathcal{G}}, P(\phi|\psi_t)$)
  $\xi^* \leftarrow$ SHORTESTPATH$(v, v_g, \widehat{\mathcal{G}})$
  $\xi_\psi \leftarrow$ ORIENTEER$(v, \widehat{\mathcal{G}}, P(\phi|\psi_t))$
  return shorter of ξ^*, ξ_ψ

Algorithm 4.3: Planning strategies.

effective in environments where the space is generally collision-free and the optimistic assumption is reasonable. In environments with many obstacles, however, dogged optimism can cause the robot to exhaustively try new paths through the same obstacle. This strategy is complete—if there exists a feasible path to the goal, D* will eventually discover it—but leveraging the posterior could help identify and rule out other paths that would pass through the same obstacle.

We compare against this baseline due to its popularity and to clarify the difference between optimistic and Bayes-aware algorithms for dynamic motion planning. The surprising effectiveness and popularity of this strategy demonstrates that even "pure" exploitation in BDMP results in additional information.

Hedged Shortest Path under Determinization (HSPD) HSPD plans on the maximum-likelihood-observation determinization [85]: only edges that are more likely to be collision-free are preserved (i.e., $P(\phi(e) = 1|\psi_t) \geq 0.5$). The shortest path from the current vertex to the goal on this determinization is deemed the "exploitation" path ξ^*. The exploitation path may be longer than the true shortest path or may not exist at all, depending on which edges the determinized roadmap is able to preserve. However, following this path and discovering a collision proves that the determinization was inconsistent with the true unknown environment, reducing the space of possible environments by a factor of 2.

HSPD also computes a second "exploration" path ξ_ψ, via ORIENTEER. Assuming that following the path would yield the maximum-likelihood collision-free observations, ξ_ψ is the shortest path that gathers enough information to reduce the space of possible environments by a factor of 2. Although each edge preserved by this determinization is more likely than not to be collision-free, traversing that path still accumulates a reduction in the space of possible environments that corresponds to how likely the path was to be in collision. Computing the exact reduction requires a posterior update along every edge of the exploration path, with additional posterior updates to consider alternative exploration paths.

HSPD hedges between ξ^* and ξ_ψ by following the shorter of the two paths. Following ξ^* to completion means that the goal has been reached. Following either ξ^* or ξ_ψ into a collision means that the maximum-likelihood-observation determinization was inconsistent, reducing the space of possible environments by a factor of 2. Following ξ_ψ to completion also reduces the space of possible environments by a factor of 2 by construction. Since each iteration shrinks the possible environment space by at least half, HSPD reaches the goal within a logarithmic number of iterations. Qualitatively, HSPD follows the exploration path until it becomes too costly to achieve the desired $2\times$ uncertainty reduction (i.e., the exploitation path to the goal becomes shorter). On a small grid-based Bayesian Canadian Traveler's Problem, Lim et al. [66] finds that an additional hyperparameter that artificially stretches the length of ξ_ψ is critical to reduce hedging-induced over-exploration.

Dynamic Replanning with Posterior Sampling (DRPS) DRPS determinizes according to a random sample from the current posterior distribution. Posterior sampling explores the space of currently plausible environments; as the posterior concentrates around the true environment, environments sampled from the posterior concentrate in the same way. Then, DRPS plans the shortest path from the current vertex to the goal in the sampled environment.

Ideally, a BDMP algorithm would explore the space of optimal paths rather than the space of environments—the objective is to minimize the total distance traversed, not to reduce uncertainty. For example, an algorithm focused on reducing environment uncertainty may continue to explore even after all plausible environments share the same optimal path. Though directly exploring the combinatorially large space of optimal paths is challenging, DRPS accomplishes this by sampling an environment and planning the optimal path in that environment. This procedure means that DRPS explores the space of plausible optimal paths $\hat{\xi}_{t+1} \sim P(\xi^*|\psi_t)$, a strategy that

balances exploration in path space using the same posterior distribution that reflects its current uncertainty. Thus, DRPS naturally avoids over-exploration without additional hyperparameters.

4.2.2 *Posterior Queries and Updates*

HSPD and DRPS assume different interfaces to the posterior distribution. HSPD requires the marginal posterior distribution

$$P(\phi(e) = 1|\psi_t) \propto \int_{\phi} P(\phi(e) = 1|\phi)P(\phi|\psi_t), \tag{4.1}$$

which must be normalized to perform the MaxLikelihood determinization. However, computing the partition function to normalize this distribution is intractable for many posteriors of interest. In contrast, DRPS requires only that the posterior distribution is sampleable: $\phi \sim P(\phi|\psi_t)$. Posterior sampling ensures that only statistically plausible environments are sampled, while algorithms that consider marginal collision probabilities effectively take a weighted average across all plausible environments.

Both HSPD and DRPS require the posterior to be updated at the end of each iteration (Algorithm 4.1, Line 5). This is only partially true for D*, which does not require a full posterior update as long as newly-discovered edge blockages are reflected in the determinization. Additionally, HSPD performs several posterior updates as part of the Orienteer step to estimate the amount of information gained by following each hypothesized exploration subpath. This is more efficient than solving the original BDMP because each posterior update assumes the determinized maximum-likelihood observations [85], but each update can still be computationally expensive for complex posterior distributions. By shifting the burden of exploration to the posterior sampling determinization strategy, DRPS can plan on the determinized roadmap using a naïve shortest path algorithm like A*. DRPS avoids further posterior updates because it does not need to consider different exploration paths in the planning step.

4.2.3 *Related Problems to BDMP*

BDMP is closely related to the Blindfolded Traveler's Problem (BTP) [96] and the Bayesian Canadian Traveler's Problem (BCTP) [66]. However, the BTP focuses on approximate posterior distributions derived from contact feedback, while the BCTP and BDMP assume that the posterior distribution is given. The BCTP automatically reveals all edge collision statuses upon reaching an incident vertex, while the BTP and BDMP require the planner to explicitly choose an edge to sense for collisions. (D*-based algorithms support a more

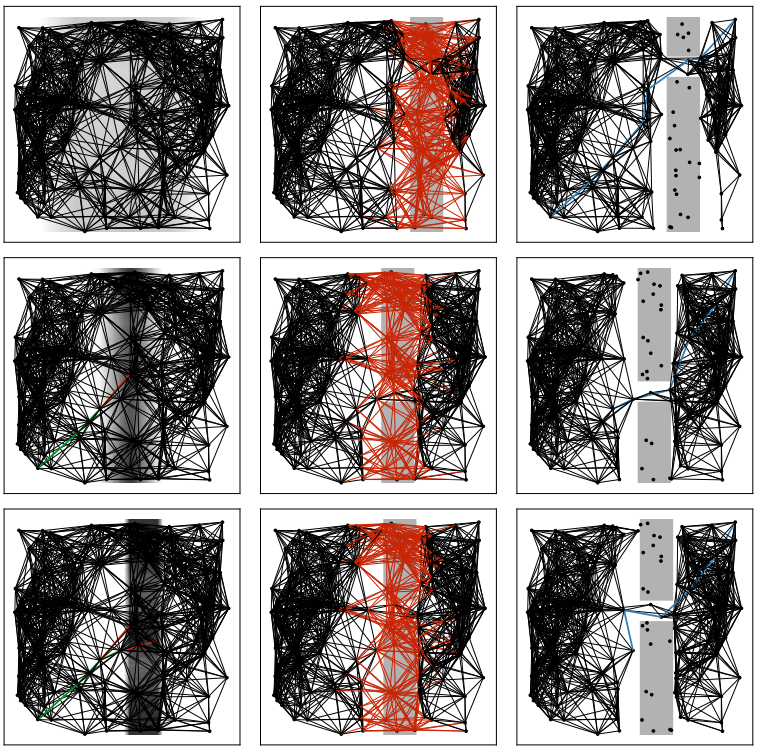

Figure 4.2: Illustrative sample run of DRPS. DRPS samples an environment from the posterior and plans the shortest path to the goal (blue). The path traverses through a region of uncertainty and eventually results in a collision (red). It determinizes again and plans a new path, which makes progress but results in another collision. However, this exploration has concentrated the posterior around the true environment; the next iteration of posterior sampling and planning reaches the goal.

general form of edge blockages, which may be discovered anywhere in the graph. In practice, however, edge blockages are often discovered within a radius of the robot's position.)

Together, these slight differences will help characterize how the proposed algorithms balance exploration and exploitation without confounding factors, i.e., with posterior approximation error (for BTP) or with varying amounts of information gained based on graph connectivity (for BCTP).

4.3 *Experiments*

We implement the three baseline algorithms (D*, HSPD, and DRPS), Bayesian dynamic motion planning environments, and posterior distribution with Python and NumPy. HSPD and DRPS share the same posterior implementation, although querying the marginal posterior distribution versus sampling from the posterior distribution is necessarily slightly different. The code has been reasonably optimized for performance while preserving the modularity necessary to evaluate different algorithms. We evaluate the algorithms on a 3.6-GHz Intel Core i7 processor with 64 GB of memory.

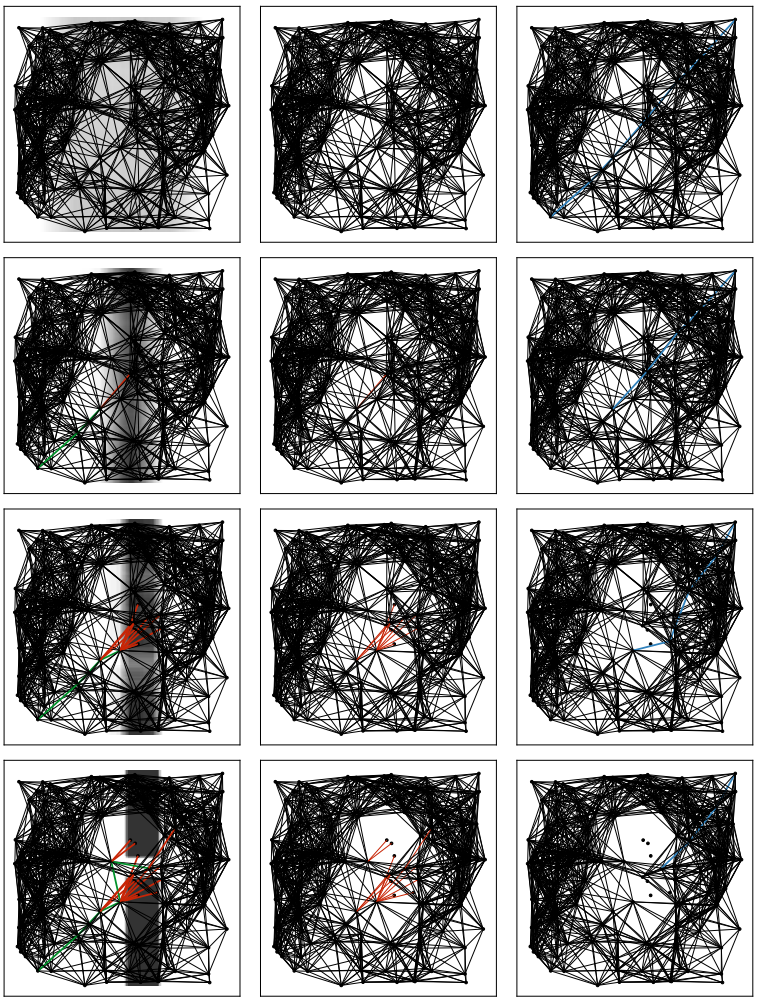

Figure 4.3: Illustrative sample run of D*. D* optimistically determinizes the posterior and plans the shortest path to the goal (blue). Due to this optimism, D* continues to plan paths through the middle region, resulting in frequent collisions. However, it eventually reaches the goal.

We evaluate planning performance on a wide variety of 2-DOF point robot environments [13] and cluttered 7-DOF Baxter manipulator environments (Figure 4.4) [41]. The smaller point robot experiments let us comprehensively evaluate how these algorithms differ and qualitatively inspect their behavior across seven datasets. The Baxter experiments are important for demonstrating the efficacy of our approach in higher-dimensional planning problems with larger roadmaps. We evaluate the algorithms on 200 problems per dataset, for a total of 1600 problems. The roadmaps for the point robot problems range in size from 100–200 vertices and 2000–5000 edges, while the roadmap for the Baxter problems contains 5000 vertices and 140000 edges. Following Lim et al. [66], we randomly generate BDMP problems based on a dataset of possible template environments. HSPD has not been previously evaluated on these

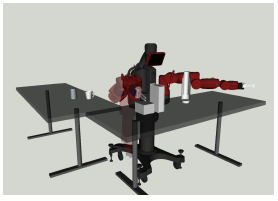

Figure 4.4: Example problem from the 7-DOF Baxter manipulator dataset, where the right arm must move from below the table to above.

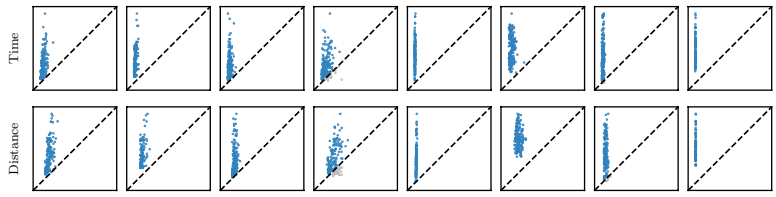

Figure 4.5: Pairwise comparisons with D* (gray). DRPS dominates D* in both planning time and distance traveled on most planning problems. DRPS performance is clustered narrowly along the x-axis, demonstrating relatively consistent planning time and distance traveled across problems within each dataset.

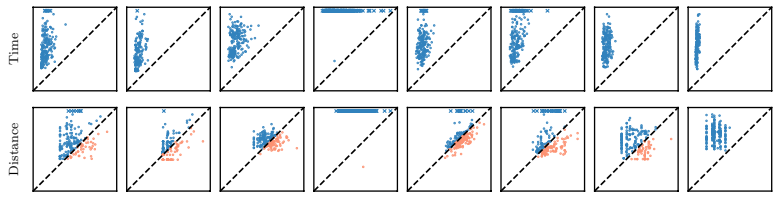

Figure 4.6: Pairwise comparisons with HSPD (orange). Planning failures are labeled with an x; HSPD fails on nearly all planning problems in the MovingWall dataset (column 4). See Figure 4.8 for an illustrative example. DRPS consistently outperforms HSPD in terms of planning time. The even spread of points about $y = x$ for the 2D dataset distance plots shows that the two algorithms achieve comparable performance in this simpler setting. However, the distance plot for the 7D Baxter dataset (bottom right) shows that DRPS consistently travels shorter distances in this more difficult setting. (Figure 4.6 visualizes the same DRPS data as Figure 4.5, but zoomed in to compare DRPS and HSPD more effectively.)

challenging environments (both in 2D and 7D); for more challenging problems, we find that HSPD can fail as a consequence of its maximum-likelihood-observation determinization strategy.

Figure 4.5 and Figure 4.6 visualize the performance of DRPS relative to D* and HSPD, respectively, on the key metrics of total planning time expended and total distance traveled. We summarize this data quantitatively in Table 4.1, which includes additional information about the number of iterations to solve each problem. DRPS travels less distance than D* and typically requires less planning time to reach the goal. This demonstrates the value of taking a Bayesian, rather than an optimistic, approach to dynamic motion planning. We see especially significant gains in the Maze 2D and the Baxter environments, where the D* optimistic assumption is frequently violated, requiring many iterations of replanning. Figure 4.2 and Figure 4.3 visualize snapshots as DRPS and D* solve the same Bayesian dynamic motion planning problem. D* expends significant travel distance trying to pass through regions that are already unlikely to be collision-free.

On most 2D problems, we find that DRPS travels a comparable distance to HSPD while spending less computation time. While HSPD occasionally failed to solve some problems in other datasets, it especially struggled to solve problems in the MovingWall dataset. We visualize snapshots of its progress in Figure 4.8, which shows that the maximum-likelihood-observation determinization is too strict: edges that are crucial for connecting to the goal are eliminated because they are likely to be in collision. However, both exploration and exploitation paths are planned on this poor approximation to the BDMP; when edges are eliminated by determinization, HSPD cannot plan either path. Furthermore, HSPD's existing exploration param-

eter cannot help it recover from this scenario. This suggests that maximum-likelihood-observation determinization may be unsuitable for motion planning settings that are likely to be in collision.

We observed the largest improvement from HSPD to DRPS on the 7D Baxter environment, both in planning time and distance traveled. We also measured the number of iterations required by each algorithm, to assess whether the root cause of the difference in planning time was due to the individual expense of each iteration or the accumulated cost of a larger number of iterations. We find that both are true: an iteration of HSPD takes $5\times$ longer than an iteration of DRPS, and HSPD requires about $4\times$ the number of iterations to reach the goal configuration. The relative difference in time spent on each iteration is likely due to the HedgedShortest planning algorithm, which requires many posterior updates to plan an exploration path. The relative difference in iterations is likely because HSPD explores more than necessary. Figure 4.7 shows that HSPD may incur additional travel distance as it continues to explore and reduce uncertainty. We conclude that BDMP algorithms must be accurately tuned for exploration. Otherwise, explicitly navigating the exploration-exploitation tradeoff during planning will incur additional computational expense without a corresponding improvement in distance traveled.

4.4 Discussion

DRPS is an efficient determinization-based strategy that carefully considers when and how expensive computations with the Bayesian posterior are performed. We analyze how other algorithms within this framework navigate the exploration-exploitation tradeoff inherent to BDMP. Experimentally, shifting the burden of exploration from planning to determinization significantly reduces total planning time—from 4–$7\times$ on 2D planning problems to $18\times$ on 7D Baxter manipulator problems. This is generally accompanied by a small improvement in total distance traveled for 2D problems and a 40% improvement in 7D problems.

From a practitioner's standpoint, the main task is defining the posterior distribution for BDMP. We believe this is an open representation learning problem: how should one estimate and infer uncertain environments? As long as that distribution supports random sampling, DRPS is simple to implement and free of tuning parameters that manually control the exploration-exploitation tradeoff. Thus, developing generative posterior models of the robot's environment offers an exciting avenue for future work. While it expends additional computation per iteration relative to popular D*-based optimistic

Metric	D*	HSPD	DRPS
OneWall			
Time (ms)	44.3 ± 3.9	45.7 ± 2.2	7.4 ± 0.7
Distance	6.2 ± 0.5	2.6 ± 0.1	2.1 ± 0.1
Iterations	20.6 ± 1.7	6.3 ± 0.2	5.2 ± 0.4
Success Rate	200 / 200	193 / 200	200 / 200
TwoWall			
Time (ms)	172.5 ± 12.8	85.4 ± 4.9	13.6 ± 1.1
Distance	6.5 ± 0.4	2.0 ± 0.1	2.0 ± 0.1
Iterations	30.7 ± 2.0	4.1 ± 0.2	2.9 ± 0.2
Success Rate	200 / 200	199 / 200	200 / 200
Forest			
Time (ms)	191.3 ± 16.8	124.3 ± 4.6	26.7 ± 2.2
Distance	6.6 ± 0.5	2.3 ± 0.0	2.2 ± 0.1
Iterations	30.7 ± 2.7	7.5 ± 0.1	4.5 ± 0.3
Success Rate	200 / 200	200 / 200	200 / 200
MovingWall			
Time (ms)	81.7 ± 7.7		26.9 ± 2.4
Distance	5.5 ± 0.4		3.2 ± 0.1
Iterations	19.7 ± 1.8		6.6 ± 0.5
Success Rate	200 / 200	1 / 200	200 / 200
Maze			
Time (ms)	999.7 ± 58.6	92.6 ± 3.5	20.2 ± 1.1
Distance	39.0 ± 2.6	3.0 ± 0.1	3.1 ± 0.1
Iterations	209.1 ± 13.9	7.2 ± 0.2	5.7 ± 0.2
Success Rate	200 / 200	188 / 200	200 / 200
Baffle			
Time (ms)	215.2 ± 11.9	105.2 ± 4.9	22.8 ± 1.8
Distance	14.3 ± 0.4	3.1 ± 0.1	3.2 ± 0.1
Iterations	57.8 ± 1.8	6.3 ± 0.2	5.3 ± 0.4
Success Rate	200 / 200	169 / 200	200 / 200
Bugtrap			
Time (ms)	246.0 ± 23.7	102.2 ± 4.2	13.6 ± 1.0
Distance	14.2 ± 1.2	2.9 ± 0.1	2.3 ± 0.1
Iterations	60.8 ± 5.7	6.3 ± 0.2	2.9 ± 0.2
Success Rate	200 / 200	200 / 200	200 / 200
Baxter (7D)			
Time (s)	100.4 ± 5.1	15.4 ± 0.6	0.9 ± 0.1
Distance	743.6 ± 29.4	28.6 ± 0.6	11.8 ± 0.4
Iterations	374.9 ± 14.7	8.2 ± 0.2	2.4 ± 0.1
Success Rate	200 / 200	200 / 200	200 / 200

Table 4.1: Online Bayesian dynamic motion planning performance on 2D [14] and 7D [41] datasets. We report the 95% confidence interval on the mean for planning time, total distance traveled, and number of iterations. Only successful planning trials are included in these confidence intervals. DRPS and D* succeeded on all planning problems. We have intentionally omitted the performance of HSPD on the MovingWall dataset; it failed to solve 199 of 200 problems. Planning times are reported in milliseconds for 2D problems and in seconds for 7D problems.

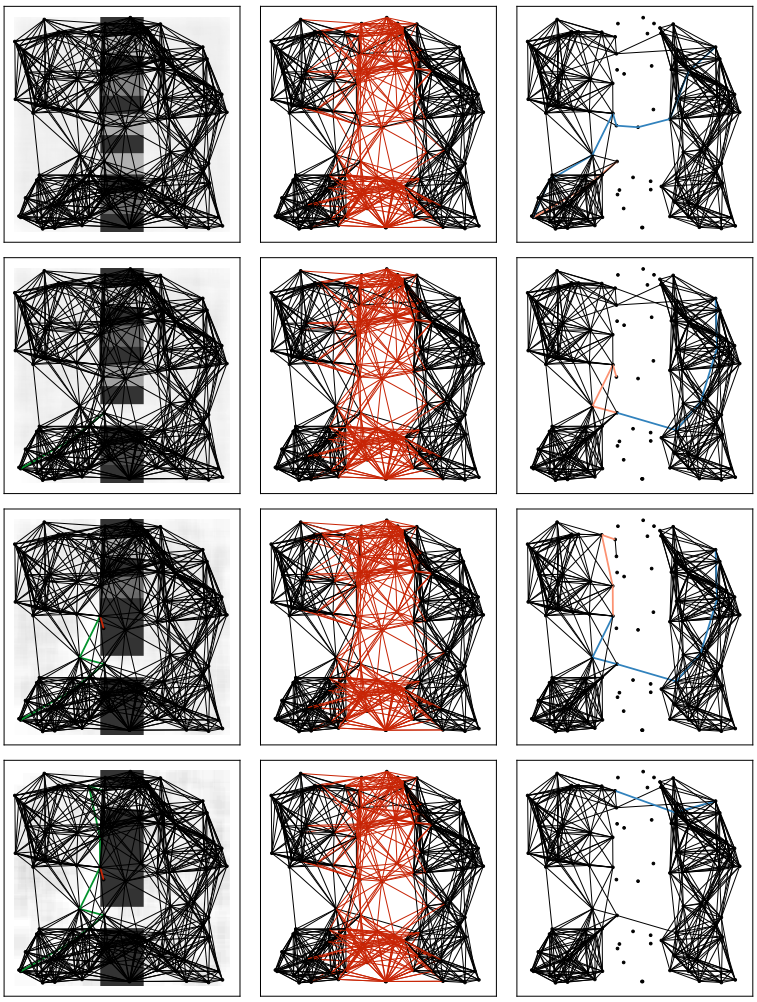

Figure 4.7: Illustrative successful run of HSPD. Given the maximum-likelihood-observation determinization of the posterior distribution, HSPD plans an exploitation path to the goal (blue) and an exploration path that explores the center obstacle region (red). Following the shorter exploration path results in a collision (red), causing an updated posterior and determinization. HSPD plans an exploitation path that backtracks and passes through the passage that is likely to be collision-free. However, HSPD instead follows the shorter exploration path (red) to completion and updates its determinization. At this final iteration, it plans only an exploitation path since there is no suitable exploration path.

strategies (to perform a full posterior update), DRPS effectively leverages information from the updated distribution to quickly plan paths through uncertain environments.

In each iteration, DRPS aims to sample from the combinatorially large space of plausible optimal paths to the goal. Because directly sampling from path space is difficult, DRPS first samples from the space of plausible environments and then plans for the optimal path in that environment. However, progress may stall if the sampled environment does not contain any paths to the goal. Even when the lack of a path plausibly reflects the posterior distribution, this behavior may be undesirable: querying a motion planning algorithm implicitly makes an optimistic assumption that a path to the goal exists, and the algorithm is tasked with finding it. Focusing sampling and planning effort on environments where a path to the goal exists presents an interesting future research direction.

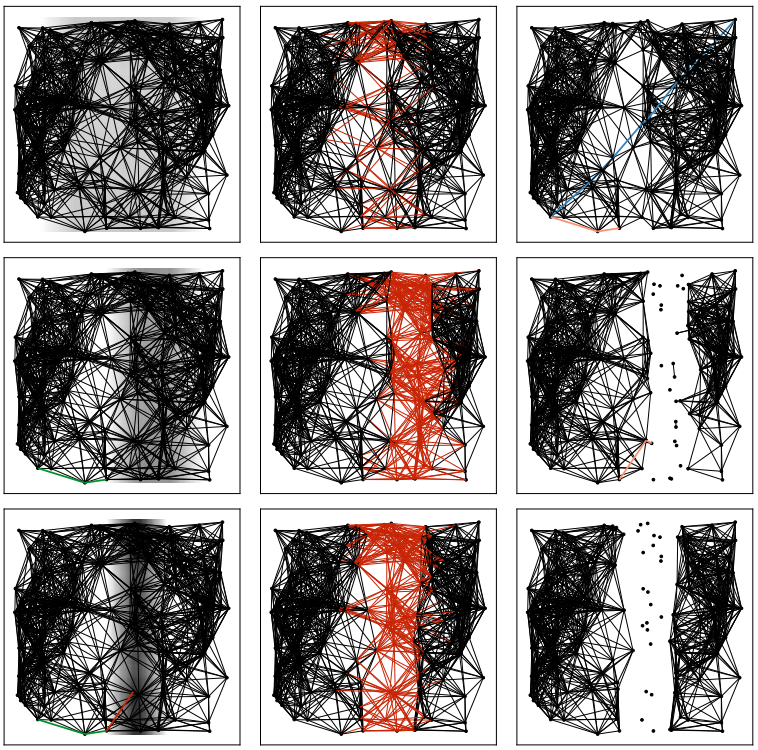

Figure 4.8: Illustrative unsuccessful run of HSPD. On the same environment as Figure 4.2 and Figure 4.3, HSPD plans on the maximum-likelihood-observation determinized graph. It follows the shorter exploration path, which updates the posterior distribution. However, the determinization eliminated all the edges that would connect to the goal because they are each *individually* unlikely to be collision-free. While HSPD continues to follow an exploration path, it eventually fails to plan either an exploitation path to the goal or an exploration path that reduces uncertainty. It is limited to the edges available in the maximum-likelihood-observation determinization.

DRPS does not explicitly optimize for combined planning and execution time, i.e., the wall clock time for the robot to reach the goal. (We measured the two components separately and used total distance traveled as a proxy for the latter.) We observe that there is an implicit tradeoff between planning and execution time. Intuitively, if a robot moves slowly, expending additional planning time to propose a shorter path $\hat{\xi}$ may be a worthy tradeoff. The Generalized Lazy Search framework introduces the concept of a toggle between different components of planning [72], which can be extended to toggle between planning and execution. Indeed, the Bayesian lazy motion planning problems considered by prior work, which permit collision-checking anywhere in the environment, define one end of the spectrum where the robot can teleport [14; 41]. Future work that introspects both the planning algorithm and underlying robot platform will be necessary to achieve this gold standard of minimizing wall clock time.

5
Probabilistic Roadmaps Under Uncertainty

This chapter is adapted from Lambert et al. [61].

The previous two chapters have considered the question of efficient Bayesian search: how should a robot plan through an uncertain environment? Within the sampling-based motion planning paradigm, however, search efficiency and path quality are ultimately contingent on the quality of the discrete roadmap approximation to the continuous configuration space. This chapter revisits that approximation: we posit that environment uncertainty should *drive* the roadmap sampling process. Leveraging this information earlier in the sampling-based motion planning pipeline can help generate roadmaps that can be searched even more efficiently to yield near-optimal paths.

5.1 Efficient Search via Sparse Roadmaps

Probabilistic roadmaps approximate a continuous configuration space with a discrete set of sampled configurations [54; 53]. Theoretical bounds have been developed for the number of uniform or low-discrepancy samples required to ensure near-optimal paths [53; 100; 51].[1] However, such bounds require many samples to generate a path through challenging "narrow passage" problems [43]. This yields dense roadmaps that cannot be searched efficiently at query time and may even be too large to store with limited onboard memory. This challenge is exacerbated in robot motion planning, where lazy algorithms are necessary to defer expensive edge collision-checking [5; 37; 17; 33; 72].

[1] A roadmap is any discrete graph approximation to the continuous configuration space, regardless of how samples were generated.

Thus, the goal is to construct sparse roadmaps that preserve the connectivity of the underlying free configuration space, admitting near-optimal paths that can be computed efficiently. Two main threads of work have been explored. Sparsification methods approach this by removing samples from dense roadmaps to reduce query-time computation, while preserving the original roadmap coverage and connectivity [21; 94]. However, probabilistic occupancy maps are not leveraged to direct the sparsification process.

Our approach, Stein Variational Probabilistic Roadmaps (SV-PRMs), is best categorized within a complementary thread of work: biasing the PRM sampling distribution to focus the roadmap's approximation power on important regions of the configuration space. Our key insight is:

> *The distribution of roadmap samples should match the distribution over feasible states.*

We formulate this task as particle-based variational inference [69]. SV-PRM deterministically optimizes the samples of a sparse roadmap to efficiently cover the space. This method only requires that the distribution is differentiable, an assumption satisfied by continuous Bayesian occupancy maps [78; 87; 98] from simultaneous localization and mapping or otherwise assumed by popular trajectory optimization algorithms [88; 117; 74; 75].

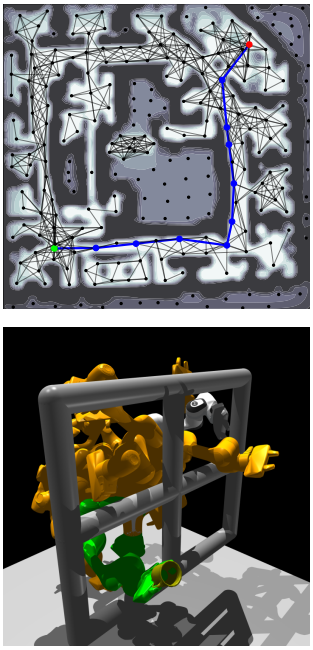

Figure 5.1: Stein Variational PRMs are generated by sampling from a posterior feasibility distribution via particle-based variational inference. Placement of roadmap vertices is governed by a gradient flow, promoting uncertainty-guided exploration of the state space during roadmap construction. This approach for generating high-quality sparse roadmaps results in efficient planning in both real-world probabilistic occupancy maps (INTEL, top) and manipulation tasks (FRANKA, bottom). In the FRANKA environment, roadmap vertices are represented by orange robot configurations, with the goal configuration shown in green.

5.1.1 *Biased Probabilistic Roadmaps*

Biasing heuristics based on the collision geometry or topology are most effective on 2D problems and increase the expense of generating a single sample [6; 39; 44; 95]. These approaches typically require additional uniform samples to provide baseline coverage of easily-sampled regions. Hybrid PRM sampling proposes an adaptive strategy for selecting among different component sampling strategies [45]. When the underlying map of the configuration space is uncertain, rejection sampling can be used to focus sampling in regions that are likely to be collision-free [73]. Recent work has trained neural networks to propose samples around predicted shortest paths [47; 86] or bottleneck nodes [59]. However, these approaches primarily focus on the single-query setting, i.e., with a specific start and goal configuration pair (x_s, x_g). Critical PRMs are sparse roadmaps designed for the same multi-query setting targeted by SV-PRMs [48]. This approach identifies nodes in a dense PRM with high betweenness centrality and trains a neural network to predict such critical nodes from local environment features. When constructing the roadmap, the algorithm densely samples configurations and includes them with probability proportional to their predicted criticality. Neural network-based approaches require a dataset of similar planning environments, while SV-PRMs perform variational inference with a differentiable probabilistic feasibility function.

A key question with biased samplers is the choice of environment representation. This is often in the form of an occupancy map [59; 95], point cloud [86], or other complete representation [47]. Explicitly constructing these representations in configuration space can

be challenging for many robotics settings, especially with partial observability. In contrast, classical geometry heuristics only require access to a binary collision checker to evaluate candidate samples [6; 39; 44], while Critical PRMs use local features to predict criticality [48]. Similarly, SV-PRMs rely only on access to a differentiable probabilistic feasibility function, and avoid explicitly evaluating the function over the entire state space.

5.1.2 *Motion Planning with Continuous Costs*

Although our work focuses on the sampling-based motion planning paradigm, other optimization methods can be leveraged to plan robot motions. Trajectory optimization methods consider the single-query setting for a specified cost function. CHOMP performs covariant gradient descent to optimize a collision-free path where obstacles are represented with smooth signed distance functions [88; 117]. TrajOpt uses sequential quadratic programming to optimize a collision-free path where obstacles are represented with convex objects [97]. Gaussian Process Motion Planning (GPMP) algorithms formulate the problem of optimizing a smooth continuous-time trajectory as probabilistic inference, which can be represented as a sparse factor graph and solved efficiently [74; 75]. However, trajectory optimization methods are sensitive to initialization. While multiple initializations or stochastic optimization methods may help discover better local minima [52; 55], these approaches do not replace complete motion planners.

Cost functions can be leveraged to bias sampling within a probabilistically complete sampling-based motion planning algorithm. Biasing heuristics based on deterministic collision geometry can be adapted for the probabilistic collision setting [73]. T-RRT is a single-query method that uniformly samples new configurations to extend the RRT, but introduces a transition test to accept an edge to a candidate configuration depending on its relative cost to the nearest node in the tree [49; 19]. Cost decreases are always accepted while higher increases have lower probability of being accepted, thus biasing the growth of the tree through low-cost regions and slowing growth through high-cost regions. For Gaussian process cost functions, GP-PRMs focus on efficiently evaluating whether roadmap edges meet a desired safety cost threshold via adaptive discretization [116]. Markov Chain Monte Carlo methods have been applied to sample uniformly from the sublevel set of a kinodynamic cost function, as the inner loop of an asymptotically-optimal planning algorithm [113]. Generating a single sample becomes computationally expensive, although this may be a necessary tradeoff for that setting. Rather than

heuristically growing a search tree or sampling uniformly from a sublevel set of the cost, SV-PRMs perform particle-based variational inference to match the cost distribution with roadmap samples. SV-PRMs only assume that the cost is differentiable and are otherwise agnostic to the cost function representation.

5.2 *Posterior-Guided Roadmap Construction*

We can represent the probability of a given configuration being collision-free with the feasibility likelihood $p(z = 1|x;\phi)$. The occupancy indicator variable $z \in \{0,1\}$ labels a given location as being in collision ($z = 0$) or collision-free ($z = 1$). Using Bayes' rule, we can obtain a posterior probability over collision-free space:

$$p(x|z = 1;\phi) = \eta p(z = 1|x;\phi)p(x) \tag{5.1}$$

where $p(x)$ is a prior probability and η is a normalizing factor.[2] To efficiently cover the free configuration space with roadmap vertices, we desire a sampling distribution that has high probability in $\mathcal{X}_{\text{free}}$ and low probability elsewhere.

[2] We use this notation for convenience, maintaining the following equivalence: $p(z = 1|x;\phi) = p(\phi(x) = 1|x;\phi)$, where $\phi(x)$ denotes whether a robot configuration $x \in \mathcal{X}$ is collision-free in environment ϕ.

This formulation bears particular significance in the partially-observable setting, where we do not have access to the ground truth occupancy labels z, but can make measurements using a probabilistic model accounting for sensor noise. In this case, the likelihood distribution $p(z = 1|x;\phi)$ can be interpreted as a probabilistic map of unoccupied regions [107]. However, such feasibility likelihoods may also correspond to negatively-exponentiated costs found within the context of trajectory optimization. In both cases, we can derive a target posterior distribution to inform a sampling scheme for PRMs.

Bayesian Occupancy Maps Bayesian inference provides a useful framework for integrating observations and incorporating uncertainty to construct probabilistic occupancy maps. The environment parameter ϕ can be obtained by performing Bayesian inference using collected occupancy data. Given a set of state-measurement pairs $x_{1:M}, z_{1:M}$, a likelihood model $p(z = 1|x,w)$, and prior $p(w)$, the posterior over model parameters can be expressed as

$$p(w|x_{1:M}, z_{1:M}) = \eta \prod_{m=1}^{M} p(z_m|x_m, w)p(w) \tag{5.2}$$

where η is a normalizing factor. The feasibility likelihood can then be modeled for any point in the configuration space $x \in \mathcal{X}$, where the parameter ϕ can be chosen as a sufficient statistic from the fitted posterior, e.g.,

$$\phi = \mathbb{E}_{p(w|x_{1:M},z_{1:M})}[w] = \mu_w. \tag{5.3}$$

Approximating the posterior over latent environment parameters is non-trivial, but a variety of scalable approaches exist. For example, Bayesian Hilbert Maps model the likelihood of free-space using the sigmoid:

$$p(z = 1|x, w) = \left(1 + \exp(w^\top \Phi(x))\right)^{-1} \tag{5.4}$$

with a feature vector $\Phi(x)$ of radial-basis functions [98].

Feasibility Distributions in Motion Planning Trajectory optimization aims to compute the optimal trajectory ξ^* that minimizes an objective functional $w(\xi)$. The solution must be feasible and avoid collisions with obstacles; this condition can be imposed by including an additional obstacle penalty term w_{obs} in the objective [88; 117]:

$$w(\xi) = w_{\text{task}}(\xi) + w_{\text{obs}}(\xi) \tag{5.5}$$

where $w_{\text{task}}(\xi)$ includes all other task objectives and constraints, such as distance-to-goal and penalty on joint-limit violations. The objective is minimized via a Gauss-Newton form of iterative gradient descent, leveraging both gradient and local curvature of the cost function for rapid convergence [88; 97; 75].

The obstacle cost at a given state $x \in \xi$ can be modeled using a truncated signed-distance field (t-SDF), which penalizes a state if a Cartesian point on the robot is within an ϵ-ball from the surface of the nearest obstacle. Approximating the robot surface using a collection of body spheres $s_{1:K}$, and letting $d(x, s_k)$ to be the distance from the surface of s_k to the nearest obstacle when in configuration x, a t-SDF value is defined by the hinge loss:

$$c(x, s_k) = \begin{cases} -d(x, s_k) + \epsilon & d(x, s_k) \leq \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

These costs are combined across spheres to construct an obstacle-cost vector: $\mathbf{h}(x) = [c(x, s_k)]\,\big|_{1:K}$. The total obstacle cost is defined as the scaled inner-product: $w_{\text{obs}}(x) = \alpha \|\mathbf{h}(x)\|^2$, where $\alpha > 0$. As in Mukadam et al. [74, 75], the obstacle cost at a given state can be interpreted as the negative log-likelihood function:

$$-\log p(z = 1|x; \phi) = \alpha \|\mathbf{h}(x)\|^2 + \text{const} \tag{5.7}$$

with parameter ϕ including obstacle locations and geometry. Here, the likelihood probability of a state being collision-free is modeled as

$$p(z = 1|x; \phi) \propto \exp(-\alpha \|\mathbf{h}(x)\|^2). \tag{5.8}$$

Motion planning can then be framed as a *maximum a-posteriori* inference problem:

$$\xi^* = \arg\max_{\xi} p(\xi|z = 1; \phi) = \arg\max_{\xi} \prod_{t=0}^{T} p(z = 1|x_t; \phi) p(\xi) \tag{5.9}$$

where trajectories are discretized into a set of waypoints $\xi = x_{0:T}$ over a horizon T.

Although gradient-based controllers (e.g., iLQR) and motion planners (e.g., GPMP [74; 75]) can quickly return a feasible trajectory, the solution may only be *locally* optimal. Because trajectory optimization methods are sensitive to initialization, it is desirable to discover an initialization within a basin of attraction that yields a low-cost optimized trajectory. While multiple initializations or stochastic optimization methods may help discover better basins of attraction [52; 55], sampling-based motion planning can also be leveraged to compute a good initialization. This initialization can be computed by sampling PRM vertices (e.g., from the feasibility posterior over configuration space (5.1)) and generating the shortest path on the resulting roadmap.

In both scenarios, we would like to directly sample from the posterior distributions to generate collision-free roadmap vertices. Unfortunately, obtaining a closed-form expression for this target distribution is intractable in general. The posterior sampling procedure must then be approximated using methods such as Markov chain Monte Carlo (MCMC) or variational inference (VI).

5.2.1 *Variational Inference via Stein Variational Gradient Descent*

Variational inference is a powerful tool for approximating challenging probability densities in Bayesian statistics. As opposed to MCMC methods, VI formulates inference as an optimization problem. A proposal distribution $q(x)$ is selected from a family of tractable distributions $\mathcal{Q}$ to minimize the KL divergence with the target posterior distribution $p(x|z)$

$$q^*(x) = \arg\min_{q \in \mathcal{Q}} D_{KL}(q(x) \| p(x|z)). \tag{5.10}$$

Traditional VI methods typically require careful selection of the family $\mathcal{Q}$, which is often chosen to have a tractable parameteric form at the expense of introducing bias.

Stein Variational Gradient Descent leverages a non-parameteric particle-based representation of the posterior distribution rather than committing to a parametric family $\mathcal{Q}$ [69; 112]. This approach approximates a posterior $p(x|z)$ with a set of particles $x^{1:N}$. Particles are iteratively updated with a step size ϵ according to

$$x^i \leftarrow x^i + \epsilon \delta^*(x^i). \tag{5.11}$$

The function $\delta^*(\cdot)$ lies in the unit ball of a reproducing kernel Hilbert space (RKHS). This RKHS is characterized by a positive-definite

kernel $k(\cdot,\cdot)$. The term $\delta^*(\cdot)$ represents the optimal perturbation or velocity field (i.e., gradient direction) which maximally decreases the KL divergence:

$$\delta^* = \underset{\delta\in\mathcal{H},\|\delta\|_{\mathcal{H}}\leq 1}{\arg\max} \left\{ -\nabla_\epsilon D_{KL}(q_{[\epsilon\delta]} \| p(x|z)) \right\}, \tag{5.12}$$

where $q_{[\epsilon\delta]}$ indicates the particle distribution after taking an update step. Liu and Wang [69] shows that this yields a closed-form solution which can be interpreted as a functional gradient in RKHS, and can be approximated with the set of particles:

$$\widehat{\delta^*}(x) = \frac{1}{N}\sum_{i=1}^{N} \left[k(x^i, x)\nabla_{x^i} \log p(x^i|z) + \nabla_{x^i} k(x^i, x) \right]. \tag{5.13}$$

This update rule has two terms that control different aspects of the algorithm. The first term is essentially a scaled gradient of the log-likelihood over the posterior's particle approximation. The second term is known as the *repulsive force*: intuitively, it pushes particles apart when they get too close to each other and prevents them from collapsing into a single mode.[3] This allows SVGD to approximate complex, possibly multi-modal posteriors. This optimization structure empirically provides better particle efficiency than other popular sampling procedures like MCMC [10]. The deterministic gradient-based updates result in smooth transformations of the proposal distribution, a property which makes SVGD particularly attractive for trajectory optimization and inference.

[3] In the case of a single particle, this repulsive force term vanishes because $\nabla_x k(x, x) = 0$. Then, SVGD reduces to a standard optimization of the log-likelihood (i.e., a MAP estimate of the posterior).

Hessian-Scaled Kernels The convergence and accuracy of SVGD can be significantly improved by incorporating curvature information into the kernel [18; 112]. For instance, a positive-definite matrix M can be used as a metric to scale inter-particle distances within an anisotropic radial-basis function kernel:

$$k(x^j, x^i) = \exp\left(-\frac{1}{2h}(x^j - x^i)^\top M (x^j - x^i) \right), \tag{5.14}$$

where h is the bandwidth parameter. Curvature information can then be shared across particles by averaging their local Hessian evaluations. Specifically, denoting the negative Hessian matrix to be $H(x) = -\nabla_x^2 \log p(x|z)$, we can define the metric $M = \frac{1}{N}\sum_{i=1}^{N} H(x^i)$, which is computed using x^i-values from the previous iteration.

5.2.2 *Stein Variational Probabilistic Roadmaps*

As an alternative to random or purely heuristic-driven sampling for PRMs, we propose to approximate the posterior distribution of feasible space $p(x|z=1;\phi)$ using a set of SVGD particles $x^{1:N}$.

In the *Inference* phase, particles are initialized by drawing samples from the prior distribution over configuration space: $x^i \sim p(x)$. The particle distribution is then optimized via SVGD, where at each iteration, the log-likelihood and log-prior gradients are computed for each particle in parallel:

$$\nabla_x \log p(x^i|z=1;\phi) = \nabla_x \log p(z=1|x^i;\phi) + \nabla_x \log p(x^i). \quad (5.15)$$

Given a choice of kernel $k(\cdot,\cdot)$, the RKHS gradient in (5.13) is computed for each particle. The particles are then updated, and the process is repeated until convergence.

During the *Construction* phase, the feasibility of a given point x is imposed by a chance constraint on the feasiblity likelihood: $p(z = 1|x;\phi) \geq \beta$, where $\beta \in [0,1]$. Vertices and edges are only accepted into the graph $\mathcal{G}$ if this constraint is satisfied, ensuring that they are collision-free with probability at least β. Particles which violate the chance-constraint are removed from the vertex-candidate set. Edges are connected between two vertices if the edge length is less than the connectivity radius ρ, which defines the set of nearest neighbors for each vertex. In practice, this can drastically reduce the number of edges in the graph, therefore decreasing the problem size in the planning phase.

In the *Planning* phase, a feasible path is generated by first performing a "collision check" on a candidate edge to ensure that the constraint $p(z = 1|x;\phi) \geq \beta$ holds. Equidistant query points are generated along the edge at a fixed resolution, and edge feasibility is approximated by ensuring that all query points satisfy the chance constraint. If the edge is accepted in to the graph $\mathcal{G}$, its length is evaluated and stored. Then, any roadmap shortest path algorithm can be used to generate a solution ξ^*.

A subtle but key aspect of this VI-based PRM approach is that a candidate set of vertices are *not* generated by first fitting a parameterized distribution $q(x)$ to the target posterior $p(x|z = 1;\phi)$, then sampling points from $q(x)$. Rather, the particles themselves both govern the implicit empirical distribution q and comprise the set of candidate vertices. As these vertices are *deterministically* optimized by the SVGD algorithm, they will approach regions of the space which are more likely to be collision-free, while the repulsive-force term ensures particle diversity and adequate coverage of the target distribution. The non-parametric nature of the algorithm allows this approach to be flexible and model highly multi-modal distributions—critical for covering the many modes of free configuration space.

This uncertainty-guided procedure for PRM generation is quite general, and does not necessitate sampling heuristics. However, the prior distribution $p(x)$ can be chosen to bias portions of the config-

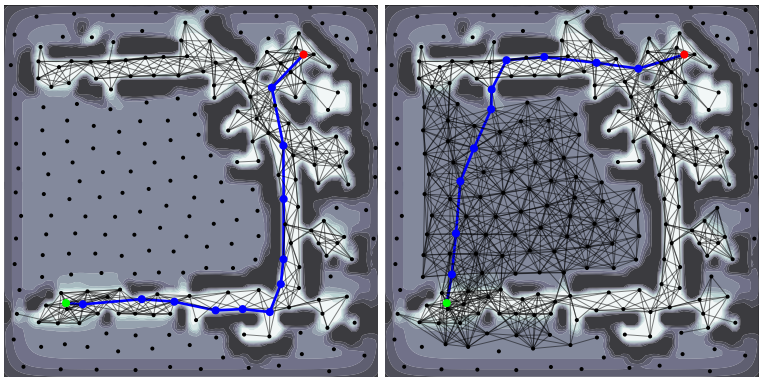

Figure 5.2: SV-PRMs in partially-explored INTEL environment, with conservative $\beta = 0.54$ (left) and optimistic $\beta = 0.45$ (right) roadmap generation.

uration space if required. Furthermore, this approach can exploit gradient-based information of the target distribution, and can be implemented in batch for efficient GPU parallelization.

With the threshold parameter β, we can control optimism in graph construction. Low values can lead to graphs which populate edges in uncertain or unexplored areas, producing optimistic path solutions with shorter paths. Conversely, higher values restrict vertices and edges to high-probability regions, resulting in longer paths. This behavior is depicted in Figure 5.2 with a partial occupancy map generated from the Intel dataset.

5.3 *Experiments*

We evaluate Stein Variational PRMs and relevant baselines on three planning environments: (1) a synthetic planar point navigation problem with challenging obstacle distributions, (2) a probabilistic occupancy map generated from a real-world baseline, and (3) a high-dimensional simulated manipulation problem. In all experiments, we use the anisotropic RBF kernel. For 2D examples with uniform priors, we use the following positive-definite metric to scale the pairwise particle distances: $M = \frac{1}{N}\sum_{i=1}^{N} \nabla_x \log p(z=1|x^i) \nabla_x \log p(z=1|x^i)^\top$.

We evaluate different roadmap generation strategies, where the traditional PRM generates samples according to a pseudorandom number generator or quasirandom sequence, potentially with cost-based rejection sampling. SV-PRM is initialized with the samples from PRM, then performs SVGD on the particle set. The connection strategy described in the previous section is used for both roadmaps. At query time, the specified start and goal configuration are connected to the roadmap. The LAZYSP [17] algorithm is then used to search the roadmap, lazily evaluating edges to check whether the cost threshold would be exceeded. We parallelize SVGD evaluations using a PyTorch implementation, allowing efficient GPU-driven inference.

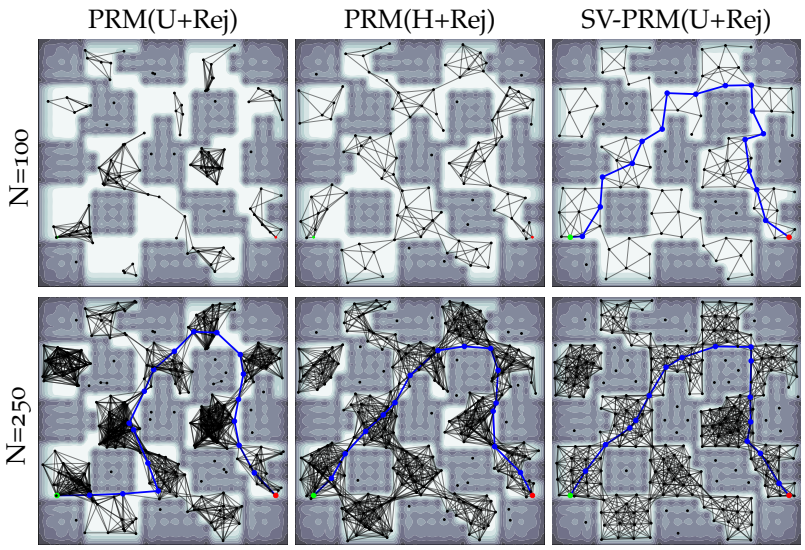

Table 5.1: CHECKBOARD environment roadmaps generated with different sampling strategies and number of samples. The planning query consists of the start (green) and goal (red), and the shortest path (blue) is visualized if it exists.

	PRM				SV-PRM			
	U	U+Rej	H	H+Rej	U	U+Rej	H	H+Rej
N = 100	2 / 30	15 / 30		✓	29 / 30	30 / 30	✓	✓
N = 250	20 / 30	30 / 30	✓	✓	30 / 30	30 / 30	✓	✓

Table 5.2: Number of successful planning trials on the CHECKERBOARD environment with N=100 and 250 sampled vertices. Uniform-based samplers are evaluated across 30 random seeds. The low-discrepancy Halton-based sampler [34] is deterministic, so success is denoted with a ✓. In the low-particle regime, uniform samples (with or without rejection) rarely capture a path between the start-goal pair. Despite these poor initializations, the SVGD optimization yields roadmaps with feasible paths in all but one trial.

5.3.1 *Point Robot Navigation*

In the synthetic CHECKERBOARD environment, we compare performance in low- and high-particle regimes across various initialization schemes. The continuous target distribution is generated by fitting a grid of RBF features to a binary occupancy map. In the real-world INTEL environment (Figure 5.1), we fit a Bayesian Hilbert Map (BHM) [98] to LIDAR data from the Intel Research Lab in Seattle [42]. The resulting probabilistic occupancy map is the SV-PRM target distribution. We introduce a barrier function to prevent particles from exceeding the map limits.

We report *query-dependent* metrics on the CHECKERBOARD environment, which measure the benefits of using SV-PRMs for specific start-goal planning queries. Uniform-based samplers are evaluated across 30 random seeds (with and without rejection), while the low-discrepancy Halton-based sampler [34] is deterministic and only requires one trial (with or without rejection). We select a challenging start-goal pair that requires the roadmap to pass through several narrow passages (Table 5.1). Table 5.2 reports the number of successful planning trials for low- and high-particle regimes, showing that SV-PRMs better capture the connectivity of the space with the same number of samples. Compared to the larger PRMs that are needed to match the smaller SV-PRMs connectivity, this improved sparsity yields faster planning times and statistically-equivalent path lengths.

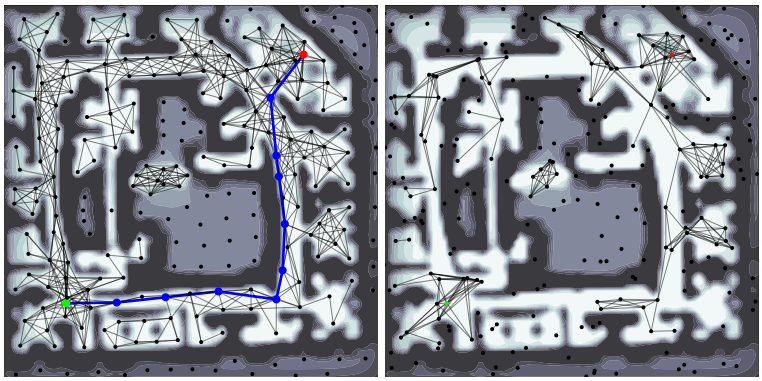

Figure 5.3: Optimized SV-PRM (left) and initial PRM (right) in fully-explored INTEL environment. SV-PRM(Uniform) has improved connectivity relative to PRM(Uniform).

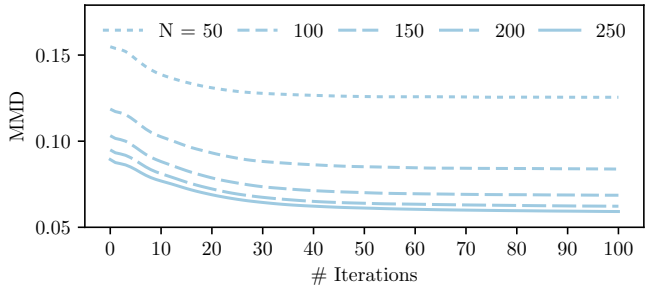

Figure 5.4: Maximum mean discrepancy (MMD) of SV-PRM(Uniform) with respect to the uniform feasible distribution of the ground truth fully-mapped Intel-BHM model (Figure 5.3). Results are averaged across 100 trials. MMD decreases with SVGD iterations, and with increasing particle number N.

On the INTEL environment, we additionally consider *query-independent* metrics that evaluate the approximation quality of the resulting roadmap without a specific planning query context. These metrics help understand the multi-query performance of SV-PRM.

First, we simulate a partial map by limiting the amount of data ingested by the BHM. Figure 5.2 visualizes the map when the BHM has consumed little data; the outer hallway loop has yet to be identified. The SV-PRM on this partial map reflects the resulting uncertainty in the model. Many vertices are in the likely collision-free spaces, but vertices are also distributed throughout the unmapped (and therefore uncertain) area. By choosing the cost threshold appropriately, SV-PRMs preserves more vertices and edges in the roadmap. The same SV-PRM can yield multiple candidate paths for a specific query by varying the threshold: a conservative threshold safely connects the start and goal via the known corridor, while an optimistic threshold navigates through the unknown and finds a shorter (but ultimately invalid) path. In this way, users are able to specify their degree of certainty in the map quality. SV-PRMs cull vertices accordingly to efficiently plan paths at the desired cost threshold. If the environment has been fully mapped, the cost should be set conservatively to guarantee safety.

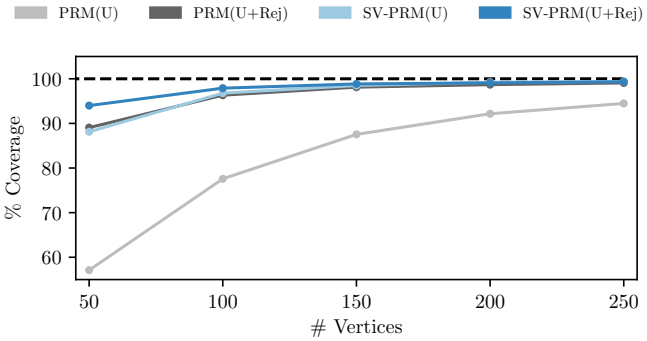

Figure 5.5: Roadmap coverage on Intel environment. SV-PRM(Uniform) and PRM(Uniform+Rejection) achieve comparable coverage, while SV-PRM(Uniform+Rejection) achieves slightly higher coverage with greater improvement in the low-particle regime.

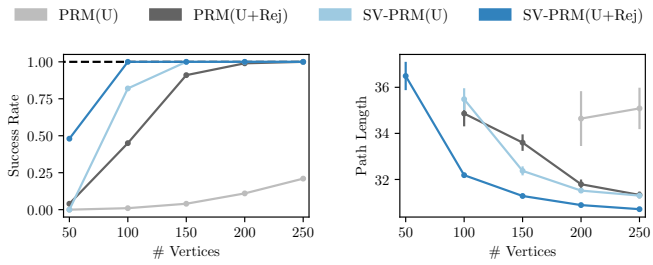

Figure 5.6: Query-dependent metrics on Intel environment across 100 random trials, as a function of vertices in the roadmap. (Left) Rate of trials with feasible solution. SV-PRM(Uniform+Rejection) succeeds on over 40% with 50 vertices and all 100 trials with 100 vertices. (Right) Mean solution path lengths. SV-PRM(Uniform+Rejection) has lower solution path length than both PRM baselines, across all roadmap sizes. SV-PRM(Uniform) achieves comparable average solution path length to PRM(Uniform+Rejection), while solving a higher rate of problems.

To further characterize this partial map setting, we evaluate the maximum mean discrepancy (MMD), which measures the difference between two distributions based on samples drawn from each of them [31]. For this environment, we compare the SV-PRM samples to a ground-truth distribution defined as the uniform distribution over collision-free states according to the fully mapped space (Figure 5.3). This differs from the partial map that SV-PRM optimizes the particles against, but demonstrates how particles have been distributed throughout the unmapped regions. Figure 5.4 visualizes the MMD as particles are optimized against the partial map, demonstrating that SVGD improves the MMD over time. Increasing the number of particles also lowers the MMD, although with diminishing returns.

Figure 5.5 demonstrates that SV-PRMs improve the configuration space coverage relative to the corresponding PRM initialization. This is estimated by sampling 1000 collision-free states in the partial Intel BHM map and attempting to connect them to roadmap vertices via collision-free edges.

Note that this coverage metric does not characterize the connectivity of the roadmap; as demonstrated in Table 5.1, rejection sampling can achieve good coverage without capturing the connectivity of the underlying space and yielding a feasible path. On query-dependent metrics, Figure 5.6 shows that both SV-PRM initializations enjoy shorter or comparable paths for the same number of vertices as the PRM baselines, while solving a higher rate of problems.

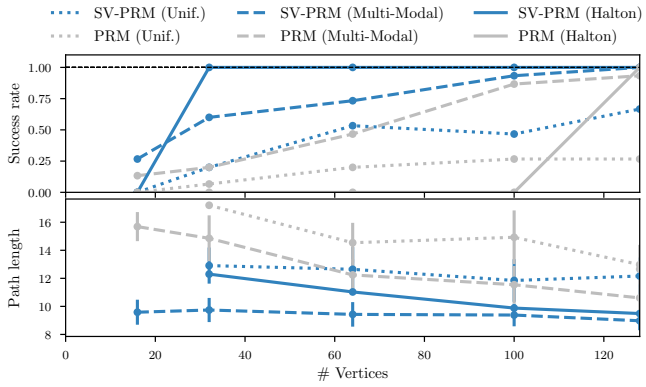

Figure 5.7: Success rate and average path length on the FRANKA manipulator reaching environment. Comparison between SV-PRM and the PRM baseline for 30 trials (10 per random seed) using different initialization distributions (i.e., priors) across varying number of vertices: $N \in \{16, 32, 64, 100, 128\}$. Particle-based posterior sampling with SV-PRM significantly improves the planning solution, particularly in the low-particle regime.

5.3.2 *Franka Manipulator Planning*

To demonstrate the scalability of SV-PRMs in higher-dimensional robotics problems, we test the approach on a simulated FRANKA reaching task (Figure 5.8). Here, the start pose of the modeled Franka arm is configured to be reaching in the top-right compartment. A feasible configuration space path must be planned to reach the goal configuration (green) in the lower left compartment, while avoiding the cabinet frame (gray). The obstacle cost $\mathbf{h}(x)$ is modeled using a smoothly varying t-SDF [117] with an offset value of 0.25m, and a feasibility likelihood of $\exp(-10\|\mathbf{h}(x)\|^2)$. The prior $p(x)$ is formulated as a uniform distribution, with exponentially-decreasing probability near the system joint-limit values.

As in the Navigation experiments, we compare our approach to traditional PRMs, varying the initialization type between pseudorandom uniform samples and quasirandom Halton samples. Furthermore, we add a task-based heuristic sampling distribution using a multi-modal mixture of Gaussians. Three isotropic Gaussian components are centered on the start and goal configurations, as well as on a retracted "home" away from the obstacles.

Figure 5.7 compares path statistics between SV-PRM and PRM for different initializations. SV-PRM manages to generate a higher degree of feasible joint-space across vertex counts for all cases, and yields lower path costs when solutions are found. Inference with SVGD optimizes the roadmap samples by pushing infeasible configurations into collision-free regions, increasing coverage of the free configuration space.

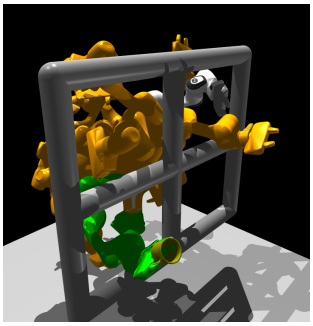

Figure 5.8: Duplicate of Figure 5.1 for easier reference. Roadmap vertices are represented by orange robot configurations, with the goal configuration shown in green.

5.4 *Discussion*

The Stein Variational Probabilistic Roadmap is a robust method for efficiently generating graphs well-suited for multi-query planning, outperforming existing biased-sampling PRM approaches. Our point robot navigation experiments show that SV-PRM yields roadmaps that admit high-quality paths even with a low sampling budget. Our FRANKA manipulator experiments show that SV-PRM scales well to higher dimensions and yields shorter paths with high success rate even in low sample regimes.

Future work will investigate how to pick the appropriate prior distributions as well as number of particles, with more emphasis on planning for manipulation as these results were very promising. Another exciting result worth investigating further is the tunable optimism parameter for partially explored environments. This is especially applicable when running the algorithm in an online setting where the particles are updated incrementally as new observations arrive, similar in spirit to the results in [98].

This approach complements work on learned neural network samplers that leverage experience to guide sampling [47; 59; 86; 48]. SV-PRMs can continue to optimize these biased samples or directly incorporate the scoring functions as a probabilistic cost.

6
Conclusion

This dissertation aims to develop efficient algorithms for motion planning with uncertainty. Pragmatic tradeoffs and approximations are integral to our approach, and we hope that stating them plainly will empower practitioners to make more informed choices about the algorithms that they deploy. Instead of hoping for a mythical master algorithm that efficiently and optimally solves all planning problems, we believe that roboticists should characterize the distribution of planning problems that they want to solve efficiently and near-optimally. The proposed algorithms have been effective for the cluttered-but-structured environments that we have considered in this dissertation. However, they are not universal replacements for other algorithms that plan with uncertainty; a careful analysis of tradeoffs is required. Our intent is to simply present them as options in the algorithmic toolbox, to be used when appropriate for the task at hand. This final chapter reviews some of these design decisions.

Sampling-Based Motion Planning Rather than planning through the continuous high-dimensional configuration space, sampling-based algorithms plan on a discrete roadmap approximation. This concedes any possibility of computing the optimal path for the benefit of quickly computing high-quality optimized paths. However, search efficiency and path quality are ultimately contingent on the quality of the roadmap approximation.

Expected Performance in Bayesian Motion Planning A Bayesian approach to motion planning is a double-edged sword, promising improved expected case performance while permitting poor worst case performance so long as the posterior distribution deems those scenarios to be improbable. This places a greater responsibility upon the practitioner to properly characterize the problems of interest: are these scenarios unrealistic pathological cases that can be overlooked, or are strong performance and efficiency still important?[1]

[1] Critically, a practitioner cannot use the presence of mathematical formalism to deflect blame for poor performance. Operationalizing uncertainty as the Bayesian posterior arguably allows these assumptions to be stated more openly and interrogated.

Bayesian Regret Minimization The proposed Bayesian search algorithms sacrifice Bayes-optimality in return for significant efficiency gains. Determinization-based approaches have well-known limitations[2]: the act of determinization eliminates uncertainty and thus subtracts any exploratory value from information-gathering actions that do not progress toward the goal. Thus, determinization-based approaches may not discover solutions that require those exploratory actions to reach the goal.

[2] Appendix B of Lee et al. [64] contains a thorough comparison of Bayes-optimality and posterior sampling-based determinization.

6.1 Future Work

6.1.1 Characterizing Uncertainty in Real-World Environments

Motion planning is an excellent testbed for understanding uncertainty without the additional complexities of stochastic transitions. We focus on full-horizon replanning, rather than mix the potentially confounding factors of receding-horizon replanning and terminal value functions. Prior work has found that even simple strategies for planning with uncertainty are effective in many environments that robots face. Indeed, the popularity of optimistic approaches shows that planning algorithms may not even need to fully consider uncertainty to be effective.

Integrating uncertainty directly into the decision-making process can itself be computationally intractable. Thus, problems that can be solved efficiently and near-optimally via determinization and replanning should not be dismissed as probabilistically uninteresting [67]. We need a deeper understanding of the settings where we can efficiently plan with uncertainty, and the settings that require more sophisticated and expensive planning algorithms. In what domains are simple heuristics like optimism sufficient, and when are Bayesian motion planning algorithms based on posterior sampling better? Given the general intractability of solving POMDPs, when would a POMDP algorithm be more suitable than posterior sampling? This dissertation builds on prior work characterizing some of these boundaries; we can view these approaches on a spectrum of probabilistic and uncertainty-aware algorithms.

Although a full POMDP formulation sets the gold standard for planning under uncertainty, our hypothesis is that many useful motion planning problems can be solved adequately without that formulation. Motion planning problems can be contrived such that POMDP formulations are required, but we need to understand what structures of uncertainty permit optimism and posterior sampling strategies to be effective in so many cases. Optimistic strategies sidestep the expense of updating the Bayesian posterior, are effec-

tive in environments where the space is generally collision-free and relatively uncluttered. POMDP formulations seem to be most effective in environments where exploration is in greater opposition to exploitation and it is difficult to learn about the environment. In the Bayesian motion planning problems considered in this dissertation, simply following a path through uncertain regions provides valuable information, rendering explicit information-gathering actions mostly unnecessary. Algorithms that maximize the Bayesian value function may result in higher expected value, but we find that the significant computational tradeoff is beneficial.

The Efficiency of Bayesian Motion Planning As part of characterizing uncertainty, we will need to narrow the gap between theoretical analysis and real performance. Our analysis that emphasizes expected performance under the Bayesian posterior rather than worst-case performance is one step in this direction. We hope that our connection between anytime motion planning and Bayesian reinforcement learning will open the door to further regret analysis of Bayesian motion planning algorithms.

However, one challenge is that these posterior-independent bounds apply to all possible posterior distributions, and are therefore quite loose when considering a specific posterior distribution. Future work will hopefully be able to characterize these performance bounds based on geometric properties of the environment and the posterior distribution. Depending on the importance of worst-case or near worst-case performance, we may also consider alternative objectives such as conditional value at risk [90; 91].

6.1.2 *Pragmatic Considerations for Planning with Posteriors*

The proposed planning algorithms have been evaluated in simulation, and we have taken care to consider high-dimensional manipulator planning problems. Our results suggest that the roadmap abstraction and offline precomputation helps these algorithms to scale to these problems. However, we anticipate that further improvements to these algorithms will be beneficial for deploying to physical robot systems.

Operationalizing Uncertainty As previously discussed, the first and most important step is to design a posterior distribution that summarizes the practitioner's domain knowledge regarding the environment and sensors. Iteratively designing and interrogating this module is important for ensuring that the distribution reflects the environments and planning problems that the robot is prepared to

solve. Our proposed search algorithms, PSMP and DRPS, have no hyperparameters; the posterior is the only component that requires tuning. Future work that helps practitioners interpret their posterior model throughout this design process will be critical. Our current approach is simply to visualize many samples from the distribution to observe whether the posterior accurately describes realistic problems.

One strategy for implementing a posterior distribution is to work directly with a sensor's local observation model. Our experiments with the Bayesian motion planning algorithms have focused on very simple observation models, while the roadmap optimization algorithms have considered more realistic LIDAR-based models. Incorporating these latter observation models as posterior distributions for planning will help make the Bayesian motion planning algorithms more widely applicable.

Explicitly characterizing the joint properties of the environment and sensor may be challenging. We believe that our approach may be fruitful for combining motion planning with machine learning.[3] Data-driven representation learning methods can characterize how raw environment observations provide information about latent environment parameters. Motion planning algorithms should exploit the structure and information of a given posterior distribution to search efficiently. We emphasize that all learned models must still be interrogated thoroughly to ensure that the distribution of planning problems is accurately described. However, we believe this separation of concerns will help practitioners solve real-world motion planning problems.

[3] Generative models of the robot's environment seem particularly well-suited for posterior sampling-based planning algorithms.

Biasing for Progress From an algorithmic perspective, both PSMP and DRPS solve shortest path queries to sample from the distribution over optimal paths.[4] However, the deterministic nature of Bayesian motion planning suggests an avenue for further focusing this distribution. For example, each iteration of PSMP aims to improve the current solution; sampling a previously-yielded path makes no improvement during that iteration.

[4] This is attractive from a computational standpoint: other Bayesian search methods such as Monte Carlo Tree Search [32] or even heuristics like QMDP [68] require several calls to the search.

Sampling from the distribution of *better* paths is challenging. Our approach indirectly models the distribution of optimal paths by modeling the distribution of environments and optimizing paths within sampled environments. This makes it easy to characterize environment structure (in the posterior distribution), but difficult to focus sampling on environments that yield better paths. A interesting alternative would be to devise a scheme for directly sampling from the distribution of possible paths below a cost threshold.

In both PSMP and DRPS, planning progress is contingent on planning a collision-free path. Progress stalls if the sampled environment

does not contain any paths to the goal; no path is attempted in that iteration. Even when the lack of a path plausibly reflects the posterior distribution, this behavior may be undesirable. Querying a motion planning algorithm is implicitly an optimistic question: we hope that there is a path to the goal. Thus, focusing sampling and planning effort on environments where a path to the goal exists presents an interesting future research direction. Depending on how difficult it is to gain information, we may also consider following partial paths that do not reach the goal to help reduce uncertainty. Planning partial paths may also help avoid unpredictable path changes in large environments, where the posterior distribution may have significant variance in distant unobserved regions.

Planning Under Mismatched Posteriors In practice, perfectly aligning the posterior distribution with the test distribution of planning problems may be quite difficult. Phan et al. [84] shows that a small error in approximate inference may cause posterior sampling to incur linear regret. Our experiments with PSMP evaluated the algorithm with the significantly mismatched nearest neighbor posterior and perfectly-aligned finite set posterior. The nearest neighbor posterior demonstrated degraded performance relative to the finite set posterior (and other search algorithms) due to overexploration, to varying degrees depending on the environment.

Although recovering from significant mismatch is impossible, making posterior sampling-based algorithms robust to some mismatch would be valuable.[5] Posterior sampling with bounded optimism may be an interesting determinization strategy for this scenario, where optimism helps offset the degree of mismatch. We may also consider the performance tradeoff between a narrow posterior with slight underexploration and a broad posterior with slight overexploration. Both approaches can be combined with more robust determinization strategies: perhaps a narrow posterior with slight underexploration can be combined with some optimism to experimentally balance exploration. Determinization strategies that optimistically combine multiple posterior samples may also be effective [115; 2; 27].

[5] Ad-hoc strategies to improve robustness may be difficult to analyze theoretically, but are still useful artifacts.

Iterative Densification with Uncertainty To guarantee convergence to the true optimal path with sampling-based algorithms, incremental densification procedures sample more vertices to improve the effective roadmap resolution. We may desire this anytime ability to improve path quality while the computational budget permits. For a fixed roadmap, the PSMP and DRPS algorithms can perform significant precomputation to improve query time efficiency. In the

incremental densification setting, the new batch of sampled vertices depends on the current best path. Without precomputation, search on the densified portions of the roadmap may be slower.

Uniform or low-discrepancy samplers are typically used to improve the roadmap approximation. We can offset some of the planning time increase by relying on SV-PRM to more efficiently place the newly sampled batch of roadmap vertices. This densified roadmap can be optimized very similarly to the original SV-PRM algorithm, although vertices and edges along the current best path must be frozen in place. For computational reasons, we may also choose to freeze all vertices and edges from the previous roadmap, i.e., only updating the new batch of samples. The corresponding kernel terms for frozen vertices must remain in the SVGD update for free vertices, effectively optimizing free vertices around frozen vertices. The upper bound on cost from the current best path can be incorporated as an informed subset constraint [29] with a barrier term in the SV-PRM task objective. However, we note that as a particle-based variational inference method, large SV-PRMs can be computationally expensive to optimize.

6.2 *Closing Thoughts*

Autonomous systems exist because of the people that build them. As roboticists, we are responsible for ensuring that those systems serve real societal needs. As algorithm designers, we must understand how practitioners will actually use the artifacts we create.

Planning with uncertainty is inextricably linked to the real world. The Bayesian methods proposed in this dissertation are a small step toward bridging the gap between theory and practice. Motivated by the challenges imposed by real-world computational budgets, we have negotiated a compromise between planning time and solution quality. We hope that this demand for urgency yields a broader spectrum of algorithms for robot motion planning with uncertainty.

Bibliography

[1] Y. Abbasi-Yadkori, J. Modayil, and C. Szepesvari. Extending rapidly-exploring random trees for asymptotically optimal anytime motion planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[2] J. Asmuth, L. Li, M. L. Littman, A. Nouri, and D. Wingate. A Bayesian sampling approach to exploration in reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, 2009.

[3] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3): 235–256, 2002.

[4] M. Bhardwaj, S. Choudhury, B. Boots, and S. Srinivasa. Leveraging experience in lazy search. In *Robotics: Science and Systems*, 2019.

[5] R. Bohlin and L. E. Kavraki. Path planning using Lazy PRM. In *IEEE International Conference on Robotics and Automation*, 2000.

[6] V. Boor, M. H. Overmars, and A. F. van der Stappen. The Gaussian sampling strategy for probabilistic roadmap planners. In *IEEE International Conference on Robotics and Automation*, 1999.

[7] S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[8] B. Burns and O. Brock. Sampling-based motion planning using predictive models. In *IEEE International Conference on Robotics and Automation*, 2005.

[9] M. Chen, E. Frazzoli, D. Hsu, and W. S. Lee. POMDP-lite for robust robot planning under uncertainty. In *IEEE International Conference on Robotics and Automation*, 2016.

[10] W. Y. Chen, A. Barp, F.-X. Briol, J. Gorham, M. Girolami, L. Mackey, and C. Oates. Stein Point Markov Chain Monte Carlo. In *International Conference on Machine Learning*, 2019.

[11] Y. Chen, S. Javdani, A. Karbasi, J. A. Bagnell, S. S. Srinivasa, and A. Krause. Submodular surrogates for value of information. In *AAAI Conference on Artificial Intelligence*, 2015.

[12] S. Choudhury, C. Dellin, and S. S. Srinivasa. Pareto-optimal search over configuration space beliefs for anytime motion planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016.

[13] S. Choudhury, S. Javdani, S. S. Srinivasa, and S. Scherer. Near-optimal edge evaluation in explicit generalized binomial graphs. In *Conference on Neural Information Processing Systems*, 2017.

[14] S. Choudhury, S. S. Srinivasa, and S. Scherer. Bayesian active edge evaluation on expensive graphs. In *International Joint Conference on Artificial Intelligence*, 2018.

[15] J. J. Chung, A. J. Smith, R. Skeele, and G. A. Hollinger. Risk-aware graph search with dynamic edge cost discovery. *International Journal of Robotics Research*, 38(2-3):182–195, 2019.

[16] B. Cohen, M. Phillips, and M. Likhachev. Planning single-arm manipulations with n-arm robots. In *Robotics: Science and Systems*, 2014.

[17] C. Dellin and S. S. Srinivasa. A unifying formalism for shortest path problems with expensive edge evaluations via lazy best-first search over paths with edge selectors. In *International Conference on Automated Planning and Scheduling*, 2016.

[18] G. Detommaso, T. Cui, A. Spantini, Y. Marzouk, and R. Scheichl. A Stein variational Newton method. In *Conference on Neural Information Processing Systems*, 2018.

[19] D. Devaurs, T. Siméon, and J. Cortés. Enhancing the transition-based RRT to deal with complex cost spaces. In *IEEE International Conference on Robotics and Automation*, 2013.

[20] D. Dey, A. Kolobov, R. Caruana, E. Kamar, E. Horvitz, and A. Kapoor. Gauss meets Canadian traveler: shortest-path problems with correlated natural dynamics. In *International Conference on Autonomous Agents and Multiagent Systems*, 2014.

[21] A. Dobson and K. E. Bekris. Sparse roadmap spanners for asymptotically near-optimal motion planning. *International Journal of Robotics Research*, 33(1):18–47, 2014.

[22] A. Dor, E. Greenshtein, and E. Korach. Optimal and myopic search in a binary random vector. *Journal of Applied Probability*, 1998.

[23] J. M. Esposito and J. N. Wright. Matrix completion as a post-processing technique for probabilistic roadmaps. In *International Workshop on the Algorithmic Foundations of Robotics*, 2016.

[24] P. Eyerich, T. Keller, and M. Helmert. High-quality policies for the Canadian traveler's problem. In *AAAI Conference on Artificial Intelligence*, 2010.

[25] D. Ferguson and A. Stentz. Anytime RRTs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[26] D. Ferguson and A. Stentz. Field D*: An interpolation-based path planner and replanner. In *International Symposium of Robotic Research*, 2007.

[27] R. Fonteneau, N. Korda, and R. Munos. An optimistic posterior sampling strategy for Bayesian reinforcement learning. In *NeurIPS Workshop on Bayesian Optimization & Decision Making*, 2013.

[28] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

[29] J. G. Gammell, T. Barfoot, and S. S. Srinivasa. Informed sampling for asymptotically optimal path planning. *IEEE Transactions on Robotics*, 34(4):966–984, 2018.

[30] M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6):359–483, 2015.

[31] A. Gretton, K. Borgwardt, M. J. Rasch, B. Scholkopf, and A. J. Smola. A kernel method for the two-sample problem. In *Conference on Neural Information Processing Systems*, 2008.

[32] A. Guez, D. Silver, and P. Dayan. Efficient Bayes-adaptive reinforcement learning using sample-based search. In *Conference on Neural Information Processing Systems*, 2012.

[33] N. Haghtalab, S. Mackenzie, A. D. Procaccia, O. Salzman, and S. S. Srinivasa. The provable virtue of laziness in motion planning. In *International Conference on Automated Planning and Scheduling*, 2018.

[34] J. H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2(1):84–90, 1960.

[35] J. Hannan. Approximation to Bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139, 1957.

[36] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[37] K. Hauser. Lazy collision checking in asymptotically-optimal motion planning. In *IEEE International Conference on Robotics and Automation*, 2015.

[38] E. Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[39] C. Holleman and L. E. Kavraki. A framework for using the workspace medial axis in PRM planners. In *IEEE International Conference on Robotics and Automation*, 2000.

[40] B. Hou and S. S. Srinivasa. Dynamic replanning with posterior sampling. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022.

[41] B. Hou, S. Choudhury, G. Lee, A. Mandalika, and S. S. Srinivasa. Posterior sampling for anytime motion planning on graphs with expensive-to-evaluate edges. In *IEEE International Conference on Robotics and Automation*, 2020.

[42] A. Howard and N. Roy. The Robotics Data Set Repository (Radish), 2003. URL http://radish.sourceforge.net/.

[43] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. In *IEEE International Conference on Robotics and Automation*, 1997.

[44] D. Hsu, T. Jiang, J. Reif, and Z. Sun. The bridge test for sampling narrow passages with probabilistic roadmap planners. In *IEEE International Conference on Robotics and Automation*, 2003.

[45] D. Hsu, G. Sánchez-Ante, and Z. Sun. Hybrid PRM sampling with a cost-sensitive adaptive strategy. In *IEEE International Conference on Robotics and Automation*, 2005.

[46] J. Huh and D. D. Lee. Learning high-dimensional mixture models for fast collision detection in rapidly-exploring random trees. In *IEEE International Conference on Robotics and Automation*, 2016.

[47] B. Ichter, J. Harrison, and M. Pavone. Learning sampling distributions for robot motion planning. In *IEEE International Conference on Robotics and Automation*, 2018.

[48] B. Ichter, E. Schmerling, T.-W. E. Lee, and A. Faust. Learned critical probabilistic roadmaps for robotic motion planning. In *IEEE International Conference on Robotics and Automation*, 2020.

[49] L. Jaillet, J. Cortés, and T. Siméon. Sampling-based path planning on configuration-space costmaps. *IEEE Transactions on Robotics*, 26(4):635–646, 2010.

[50] T. Jaksch, R. Ortner, and P. Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

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

[52] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. STOMP: Stochastic trajectory optimization for motion planning. In *IEEE International Conference on Robotics and Automation*, 2011.

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

[54] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[55] J. Kennedy and R. Eberhart. Particle swarm optimization. In *International Conference on Neural Networks*, 1995.

[56] S. Koenig and M. Likhachev. Improved fast replanning for robot navigation in unknown terrain. In *IEEE International Conference on Robotics and Automation*, 2002.

[57] J. Z. Kolter and A. Y. Ng. Near-Bayesian exploration in polynomial time. In *International Conference on Machine Learning*, 2009.

[58] J. J. Kuffner and S. M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, 2000.

[59] R. Kumar, A. Mandalika, S. Choudhury, and S. S. Srinivasa. LEGO: Leveraging experience in roadmap generation for sampling-based planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.

[60] B. Lacevic, D. Osmankovic, and A. Ademovic. Burs of free C-space: a novel structure for path planning. In *IEEE International Conference on Robotics and Automation*, 2016.

[61] A. Lambert, B. Hou, R. Scalise, S. S. Srinivasa, and B. Boots. Stein variational probabilistic roadmaps. In *IEEE International Conference on Robotics and Automation*, 2022.

[62] J.-P. Laumond, N. Mansard, and J.-B. Lasserre. Optimality in robot motion: Optimal versus optimized motion. *Communications of the ACM*, 57(9):82–89, 2014.

[63] G. Lee, B. Hou, A. Mandalika, J. Lee, S. Choudhury, and S. S. Srinivasa. Bayesian policy optimization for model uncertainty. In *International Conference on Learning Representations*, 2019.

[64] G. Lee, B. Hou, S. Choudhury, and S. S. Srinivasa. Bayesian Residual Policy Optimization: Scalable Bayesian reinforcement learning with clairvoyant experts. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021.

[65] M. Likhachev, G. J. Gordon, and S. Thrun. ARA*: Anytime A* with provable bounds on sub-optimality. In *Conference on Neural Information Processing Systems*, 2004.

[66] Z. W. Lim, D. Hsu, and W. S. Lee. Shortest path under uncertainty: Exploration versus exploitation. In *Conference on Uncertainty in Artificial Intelligence*, 2017.

[67] I. Little and S. Thiébaux. Probabilistic planning vs replanning. In *ICAPS Workshop on Planning Competitions: Past, Present, and Future*, 2007.

[68] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning*, 1995.

[69] Q. Liu and D. Wang. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *Conference on Neural Information Processing Systems*, 2016.

[70] R. A. MacDonald and S. L. Smith. Reactive motion planning in uncertain environments via mutual information policies. In *International Workshop on the Algorithmic Foundations of Robotics*, 2020.

[71] A. Mandalika, O. Salzman, and S. S. Srinivasa. Lazy Receding Horizon A* for efficient path planning in graphs with expensive-to-evaluate edges. In *International Conference on Automated Planning and Scheduling*, 2018.

[72] A. Mandalika, S. Choudhury, O. Salzman, and S. S. Srinivasa. Generalized Lazy Search for robot motion planning: Interleaving search and edge evaluation via event-based toggles. In *International Conference on Automated Planning and Scheduling*, 2019.

[73] P. E. Missiuro and N. Roy. Adapting probabilistic roadmaps to handle uncertain maps. In *IEEE International Conference on Robotics and Automation*, 2006.

[74] M. Mukadam, X. Yan, and B. Boots. Gaussian process motion planning. In *IEEE International Conference on Robotics and Automation*, 2016.

[75] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots. Continuous-time Gaussian process motion planning via probabilistic inference. *International Journal of Robotics Research*, 37 (11):1319–1340, 2018.

[76] V. Narayanan and M. Likhachev. Heuristic Search on Graphs with Existence Priors for Expensive-to-Evaluate Edges. In *International Conference on Automated Planning and Scheduling*, 2017.

[77] C. L. Nielsen and L. E. Kavraki. A 2 level fuzzy PRM for manipulation planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.

[78] S. T. O'Callaghan and F. T. Ramos. Gaussian process occupancy maps. *International Journal of Robotics Research*, 31(1):42–62, 2012.

[79] S. C. Ong, S. W. Png, D. Hsu, and W. S. Lee. Planning under uncertainty for robotic tasks with mixed observability. *International Journal of Robotics Research*, 29(8):1053–1068, 2010.

[80] I. Osband and B. Van Roy. Why is posterior sampling better than optimism for reinforcement learning? In *International Conference on Machine Learning*, 2017.

[81] I. Osband, D. Russo, and B. Van Roy. (More) efficient reinforcement learning via posterior sampling. In *Conference on Neural Information Processing Systems*, 2013.

[82] J. Pan, S. Chitta, and D. Manocha. Faster sample-based motion planning using instance-based learning. In *International Workshop on the Algorithmic Foundations of Robotics*, 2012.

[83] C. H. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoretical Computer Science*, 84(1):127–150, 1991.

[84] M. Phan, Y. A. Yadkori, and J. Domke. Thompson sampling and approximate inference. In *Conference on Neural Information Processing Systems*, 2019.

[85] R. Platt, R. Tedrake, L. P. Kaelbling, and T. Lozano-Pérez. Belief space planning assuming maximum likelihood observations. In *Robotics: Science and Systems*, 2010.

[86] A. H. Qureshi, Y. Miao, A. Simeonov, and M. C. Yip. Motion planning networks: Bridging the gap between learning-based and classical motion planners. *IEEE Transactions on Robotics*, 37 (1):48–66, 2021.

[87] F. Ramos and L. Ott. Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent. *International Journal of Robotics Research*, 35(14):1717–1730, 2016.

[88] N. Ratliff, M. Zucker, J. A. Bagnell, and S. S. Srinivasa. CHOMP: Gradient optimization techniques for efficient motion planning. In *IEEE International Conference on Robotics and Automation*, 2009.

[89] J. H. Reif. Complexity of the mover's problem and generalizations. In *IEEE Symposium on Foundations of Computer Science*, 1979.

[90] M. Rigter, B. Lacerda, and N. Hawes. Risk-averse Bayes-adaptive reinforcement learning. In *Conference on Neural Information Processing Systems*, 2021.

[91] M. Rigter, P. Duckworth, B. Lacerda, and N. Hawes. Planning for risk-aversion and expected value in MDPs. In *International Conference on Automated Planning and Scheduling*, 2022.

[92] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32(1):663–704, 2008.

[93] D. Russo and B. Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014.

[94] O. Salzman, D. Shaharabani, P. K. Agarwal, and D. Halperin. Sparsification of motion-planning roadmaps by edge contraction. *International Journal of Robotics Research*, 33(14):1711–1725, 2014.

[95] M. Saroya, G. Best, and G. A. Hollinger. Roadmap learning for probabilistic occupancy maps with topology-informed growing neural gas. *IEEE Robotics and Automation Letters*, 6(3):4805–4812, 2021.

[96] B. Saund, S. Choudhury, S. S. Srinivasa, and D. Berenson. The blindfolded robot: A Bayesian approach to planning with contact feedback. In *International Symposium of Robotic Research*, 2019.

[97] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel. Motion planning with sequential convex optimization and convex collision checking. *International Journal of Robotics Research*, 33(9): 1251–1270, 2014.

[98] R. Senanayake and F. Ramos. Bayesian Hilbert Maps for dynamic continuous occupancy mapping. In *Conference on Robot Learning*, 2017.

[99] D. Silver and J. Veness. Monte-Carlo planning in large POMDPs. In *Conference on Neural Information Processing Systems*, 2010.

[100] K. Solovey, O. Salzman, and D. Halperin. New perspective on sampling-based motion planning via random geometric graphs. *International Journal of Robotics Research*, 37(10):1117–1133, 2018.

[101] A. Somani, N. Ye, D. Hsu, and W. S. Lee. DESPOT: Online POMDP planning with regularization. In *Conference on Neural Information Processing Systems*, 2013.

[102] A. Stentz. The focussed D* algorithm for real-time replanning. In *International Joint Conference on Artificial Intelligence*, 1995.

[103] A. L. Strehl, L. Li, and M. L. Littman. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10(Nov):2413–2444, 2009.

[104] M. Strens. A Bayesian framework for reinforcement learning. In *International Conference on Machine Learning*, 2000.

[105] Z. N. Sunberg and M. J. Kochenderfer. Online algorithms for POMDPs with continuous state, action, and observation spaces. In *International Conference on Automated Planning and Scheduling*, 2018.

[106] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[107] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[108] F. Tsang, T. Walker, R. A. MacDonald, A. Sadeghi, and S. L. Smith. LAMP: Learning a motion policy to repeatedly navigate in an uncertain environment. *IEEE Transactions on Robotics*, pages 1–15, 2021.

[109] J. van den Berg, R. Shah, A. Huang, and K. Goldberg. ANA*: Anytime nonparametric A*. In *AAAI Conference on Artificial Intelligence*, 2011.

[110] K. P. Wabersich and M. N. Zeilinger. Bayesian model predictive control: Efficient model exploration and regret bounds using posterior sampling. In *Learning for Dynamics & Control Conference*, 2020.

[111] N. Wagener, C. Cheng, J. Sacks, and B. Boots. An online learning approach to model predictive control. In *Robotics: Science and Systems*, 2019.

[112] D. Wang, Z. Tang, C. Bajaj, and Q. Liu. Stein variational gradient descent with matrix-valued kernels. In *Conference on Neural Information Processing Systems*, 2019.

[113] D. Yi, R. Thakker, C. Gulino, O. Salzman, and S. S. Srinivasa. Generalizing informed sampling for asymptotically-optimal sampling-based kinodynamic planning via Markov Chain Monte Carlo. In *IEEE International Conference on Robotics and Automation*, 2018.

[114] S. W. Yoon, A. Fern, and R. Givan. FF-Replan: A baseline for probabilistic planning. In *International Conference on Automated Planning and Scheduling*, 2007.

[115] S. W. Yoon, A. Fern, R. Givan, and S. Kambhampati. Probabilistic planning via determinization in hindsight. In *AAAI Conference on Artificial Intelligence*, 2008.

[116] C. Zimmer, D. Driess, M. Meister, and D. Nguyen-Tuong. Adaptive discretization for evaluation of probabilistic cost functions. In *International Conference on Artificial Intelligence and Statistics*, 2020.

[117] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa. CHOMP: Covariant hamiltonian optimization for motion planning. *International Journal of Robotics Research*, 32(9-10):1164–1193, 2013.