Manipulation in the Blind: Motion Planning for Manipulation under Uncertainty using Contacts

Muhammad Suhail Saleem

January 2026
CMU-RI-TR-26-07

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:

Prof. Maxim Likhachev (Chair)
Prof. Jeffrey Ichnowski
Prof. Oliver Kroemer
Prof. Mehmet Dogar (University of Leeds)

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.

To my parents, Aysha Bivi and Mohamed Hussain Saleem, for their endless sacrifices and boundless love.

Abstract

Humans routinely rely on the sense of touch to better perceive the world. In environments characterized by poor lighting, occlusions, limited fields of view, or sparse visual features, contact feedback often becomes a primary source of information for perceiving the environment and successfully completing manipulation tasks. Everyday examples include locating a light switch in the dark, retrieving an item from a high shelf, or reaching for a valve at the back of a kitchen sink cabinet where visual access is severely restricted. In such settings, humans actively reason about contacts to infer both the poses of objects of interest and the environmental obstacles in the workspace. Enabling robots to exhibit similar capabilities remains a fundamental challenge in autonomous manipulation. This thesis investigates search-based planning techniques that allow robots to effectively leverage contact feedback as a sensing modality, enabling robust manipulation under uncertainty.

This work focuses on two broad classes of manipulation problems in which contact plays a critical role. The first class concerns object pose uncertainty, where precise estimation of target object pose is required to complete high-precision manipulation tasks such as charger plug insertion, pipe assembly, or other tight-tolerance manipulation problems. In these settings, even small pose errors on the order of a few millimeters can lead to failure. This problem class studies how robots can actively use contacts during execution to reduce object pose uncertainty to a level sufficient for successful task completion. The second class of problems addresses manipulation under environmental uncertainty, where the locations and geometries of environmental obstacles are unknown or only partially observable. For example, when reaching into a cluttered kitchen sink cabinet with unknown obstacles and pipes, a robot must detect contacts, infer obstacle locations, and adapt its motion accordingly in order to safely reach the target (valve). Together, these two problem classes capture a wide range of real-world scenarios in which contact-driven reasoning is essential.

Planning under object pose uncertainty naturally falls within the framework of Partially Observable Markov Decision Processes (POMDPs), which are computationally expensive to solve, particularly in continuous and high-dimensional robotic domains. This thesis presents three complementary frameworks to address this challenge. The first is an experience-based preprocessing approach designed for semi-structured environments that require strong online performance. This framework leverages solutions to previously solved, similar POMDPs to accelerate future planning queries while maintaining theoretical guarantees on solution quality. An offline database of policies is constructed and queried at execution time based on the current problem instance, enabling fast online decision-making. The second framework targets less structured domains where preprocessing is impractical. It introduces an online closed-loop planning and execution approach that employs a hierarchical representation of uncertainty. By adaptively representing and reasoning about uncertainty, this method significantly reduces planning time, making online planning and execution feasible in more complex settings. The third contribution addresses a key computational bottleneck in these settings, namely the high cost of belief space transition computations. To mitigate this issue, the thesis proposes lazy

heuristic search algorithms for POMDPs that defer expensive belief updates until they are necessary, using approximate Q-value estimators to guide search. These lazy solvers substantially reduce planning time while preserving solution quality.

For the problem class of manipulation under environmental uncertainty, this thesis develops an iterative planning and execution framework that tightly couples contact sensing, environment prediction, and motion planning. The system employs a torque-based contact detection and localization module capable of detecting contacts occurring anywhere along the robot manipulator. The history of detected contacts is used to construct a partial occupancy map of the workspace, which is then extrapolated using learned occupancy estimators. A motion planning module reasons over this estimated occupancy representation to compute actions that are likely to safely and efficiently move the robot toward the goal. The framework is evaluated in simulation and on a real UR10e manipulator across two challenging domestic tasks: manipulating a valve under a kitchen sink surrounded by pipes and retrieving a target object from a cluttered shelf.

Overall, this thesis takes a step toward *manipulation in the blind*, where robots explicitly leverage contact interactions as a primary sensing modality to plan and execute manipulation tasks under uncertainty. Rather than treating contact as a failure mode to be avoided, we treat it as an informative observation that can be actively exploited to reduce uncertainty and guide motion.

Acknowledgments

Working toward a PhD is often described as a lonely endeavor, yet my experience has been anything but lonely. I have been incredibly fortunate to be surrounded by a community of people who supported me, offered guidance, motivated me during difficult stretches, and celebrated every milestone along the way. I am deeply grateful to all my friends and colleagues. The names mentioned here represent only a small subset of a much larger group of people who have shaped this journey in meaningful ways.

I would like to begin by thanking my advisor, Prof. Maxim Likhachev. I am profoundly grateful for the opportunity to have worked with Max and hold immense respect for him as both a researcher and a mentor. He has guided me for many years, beginning in my Master's program, and has played a central role in shaping not only my research but also the way I think about problems. He was one of the primary reasons I chose to pursue a PhD, and I feel truly fortunate to have learned under his mentorship. Beyond his technical brilliance and remarkable ability to ask the right questions at exactly the right time, Max has been an extraordinarily patient and supportive advisor. He consistently encouraged me to explore ideas that genuinely interested me and stood by me as I pursued them. His calm temperament and steady demeanor created an environment where I felt trusted, challenged, and supported all at once. That balance is rare, and I am deeply thankful for it. I am also grateful for the many lighthearted moments we shared over the years, which made the journey all the more memorable.

I would also like to thank my committee members, Prof. Jeffrey Ichnowski, Prof. Oliver Kroemer, and Prof. Mehmet Dogar. I am sincerely grateful for the time, thoughtfulness, and care they invested in my work. Each of them brought a distinct perspective that challenged me to think more clearly, justify my decisions more rigorously, and refine the scope of my ideas. Our discussions, both during formal milestones and informal conversations, were invaluable in strengthening this thesis and shaping me as a researcher. I deeply appreciate their guidance and support throughout this journey.

I am incredibly fortunate to have spent my years at the Search-based Planning Laboratory (SBPL) surrounded by some of the brightest and most generous people I know. SBPL truly became my second home, and given the number of hours spent there, one could easily argue it was my first. The lab was not just a place where research happened, but a space filled with friendship, collaboration, and constant intellectual energy. Rishi Veerapaneni has been a close collaborator, a dear friend, and a mentor in many respects. He has taught me an immense amount, both technically and personally. His constant energy, spontaneous technical discussions, and excitement over new ideas made everyday research inspiring and fun. I am especially glad that I got to share an office with him throughout our PhDs. Dhruv Mauria Saxena was a wonderful collaborator during my Master's and in the early stages of my PhD. He helped me find my footing and navigate the research landscape with patience and enthusiasm. We spent countless hours brainstorming and building ideas together, and he set an exceptionally high standard for what it means to be a thoughtful and supportive collaborator.

Over the course of my time in SBPL, I have had the privilege of working alongside many friends and collaborators who made this journey both intellectually rich and personally meaningful. While this list is far from exhaustive, I would especially like to thank Alvin Zou, Fahad Islam, Gopalakrishnan Venkitachalam, Hanlan Yang, Itamar Mishani, Lai Yuan, Luca Pivetti, Manash Das, Nayesha Gandotra, Raghav Sood, Shohin Mukherjee, Tushar Kusnur, Yash Oza, and Yorai Shaoul. I am deeply grateful for their camaraderie, encouragement, and the countless conversations that shaped both my work and my time in the lab. I would also especially like to thank Ramkumar Natarajan for his friendship and for all the guidance and advice he has given me over the years on both technical and non-technical fronts. Rushing a paper from conception to submission in just a few days, working day and night together, remains one of my most cherished memories from this journey. And, of course, a special mention goes to Jerry, our friendly neighborhood mouse, who kept me company during several late-night deadline crunches.

I am equally grateful to my friends outside of the lab, who played an essential role in keeping me grounded and supported throughout this journey. In particular, Arpit Agarwal and Ashwin Khadke have been unwavering in their friendship and constant presence. Our regular hangouts, long conversations, shared dinners, baking adventures, countless movie trailers, an unreasonable number of tacos, and backpacking and hiking trips always made me feel supported and at ease, no matter how intense things became academically. Arkadeep Chaudhury brought comfort and laughter at exactly the right moments through our many discussions and debates, cooking experiments, and spontaneous frozen yogurt runs. Anahita Mohseni Kabir has been one of the kindest and most thoughtful people in my life, always looking out for me in quiet and meaningful ways. I am deeply grateful for her support and generosity. Pranav Mani made my PhD years genuinely enjoyable, and our hangouts, technical and non technical discussions, introspective reflections, and shared trips added balance and perspective to this journey. I am also grateful to Harishankar Ravishankar, Sumesh Suresh, and Dhanush Babu. The regular hangouts, badminton sessions, chai breaks, countless movies, and game nights made the latter part of my PhD especially enjoyable and memorable.

I am equally thankful to Shiva Narayan, Dhruva Rao, Nitinram Velraj, Lohita Rajesh, Sameer Dambal, Mononito Goswamy, Cecilia Morales, Swaminathan Gurumurthy, and Akshay Sharma for their friendship and encouragement over the years. A special shoutout to my friends from undergrad, Baladhurgesh Paramasivan, Mohamed Naveed, and Nanda Kishore Vasudevan, who first sparked my interest in robotics and set me on this path.

And finally, I would like to thank my family - Aysha Bivi (Mom), Mohamed Hussain Saleem (Dad), Famidha Banu (Sister), Mohamed Ibrahim (Brother-in-law), Umar Ibrahim (Nephew), and Meharjan Begum (Grandmom). From instilling in me the right values, inspiring me to work hard for the things I aspire to, and supporting me unconditionally along the way, they have done more for me than I could ever fully express. They are the foundation behind everything I have achieved and everything I am. No words can fully capture the depth of my gratitude for them.

Contents

List of Tables

List of Figures

I

Introduction

1 Introduction

1.1 Motivation

Figure 1.1: Examples of humans utilizing contact feedback for tasks like object retrieval and light switch manipulation.

For several years, the primary focus in robotics has been to enable robots to navigate environments efficiently and safely, avoiding collisions or contact with environmental obstacles. Undoubtedly, this objective has addressed a substantial class of problems in both industrial and domestic settings [1, 2]. From robot manipulators picking and placing objects on factory floors [3] to robotic vacuums autonomously cleaning homes while avoiding unstructured obstacles [4, 5], the ability to reliably operate in free space has resolved a wide array of problems. However, amidst this pursuit, one crucial sensing modality often has been underutilized: contact feedback.

Humans seamlessly employ contact feedback in everyday life to better perceive and reason about their environment [6, 7]. This is especially true in situations where visual feedback is ineffective or inadequate, such as in the presence of occlusions, poor lighting, lack of visual features or contrast, or simply when using vision is inconvenient. Through touch, we infer a wide range of properties about our surroundings, including the presence or absence of obstacles, as well as object-specific properties such as surface texture, rigidity, mass, and in some cases even dynamics. These capabilities manifest in routine interactions, such as lifting an object to assess how full or heavy it is, opening a door by sensing resistance, or evaluating the firmness of food during cooking.

Two particularly important properties that humans frequently infer through contact are **i) the pose of objects of interest** and, **ii) workspace occupancy**. For the former, consider fumbling in the dark for a light switch, finding your glasses in the morning, or reaching for an object on a high shelf. In such situations, tactile feedback provides critical information about the position and orientation of objects, allowing us to locate and manipulate them without visual input. As for workspace occupancy, consider reaching into a kitchen sink cabinet to locate a valve at the back of a shelf. In this case, one typically inserts a hand into a confined and visually occluded space. As contacts occur, they are combined with prior knowledge about the types of obstacles commonly found in such environments to infer which regions of the workspace are likely occupied. By integrating contact observations with these priors, we iteratively refine our understanding of free space and adjust our motion toward the target.

With the increasing lack of structure in the environments in which robots are being deployed [8, 9], enabling robots to use contact feedback to better perceive and reason about their surroundings is becoming increasingly important. In many such settings, robots must operate with limited, unreliable, or entirely absent visual information. This thesis aims to take a step toward *manipulation in the blind*, where robots explicitly leverage contact interactions as a primary sensing modality to plan and execute manipulation tasks under uncertainty. Specifically, we introduce search-based planning techniques that allow robots to explicitly reason about and utilize contact interactions to guide decision-making under uncertainty. Rather than treating contact as a failure mode to be avoided, we treat it as an informative observation that can be actively exploited to reduce uncertainty and guide motion.

The contributions of this thesis can be divided into two parts. The first part focuses on leveraging contact interactions to reduce uncertainty about the pose of objects of interest. Here, we study how robots can use contact information to actively localize objects in order to enable reliable manipulation. A key challenge in this setting is deciding how the robot should move so that the contact observations it acquires are maximally informative. In this part, we present three works that address this problem across different settings. The second part addresses the problem of robust manipulation in uncertain environments. In scenarios where the robot lacks accurate knowledge of the workspace obstacles, we investigate how contact feedback, together with prior knowledge about the types of obstacles typically encountered in the environment, can be used to incrementally infer environmental structure. We develop a framework that enables the robot to operate efficiently under uncertainty by leveraging contact interactions to detect and navigate around obstacles in the workspace to reach a target object or goal configuration.

Figure 1.2: (left) Examples of plugin tasks (right) A human utilizing contact feedback for localizing a port to plug in a charger.

1.2 Thesis Overview

Below, we provide a brief overview of the works included in this thesis, with a detailed exposition of them included in the chapters to follow.

1.2.1 Target object pose uncertainty

Consider a task centered around a specific target object like autonomously plugging in a charger plug into a designated port as shown in Fig. 1.2 and Fig. 1.3 (in which case the target object would be the charger port). The task inherently establishes an acceptable tolerance level of pose uncertainty. In this instance, if the position estimate deviates by more than even 2mm from the ground truth position, an open-loop execution of a plugin maneuver would be unable to successfully complete the task. Hence, before initiating the plugin maneuver, it is imperative that the robot is confident about the port's position to within a 2mm threshold.

Given initial pose uncertainty, the goal of the planning algorithms discussed in this thesis is to identify motion plans/policies for the robot, which when executed in the real world would make contact observations that reduce the pose uncertainty to below the tolerance level of the task and thereby enable the robot to complete the task. An illustration of this concept can be seen in Fig. 1.4. Here, the robot executes a sequence of maneuvers that allow it to contact the port at different locations, improving its understanding of the port pose and facilitating the successful completion of the plugin procedure.

The problem of planning for active information gathering to complete a task can be formulated

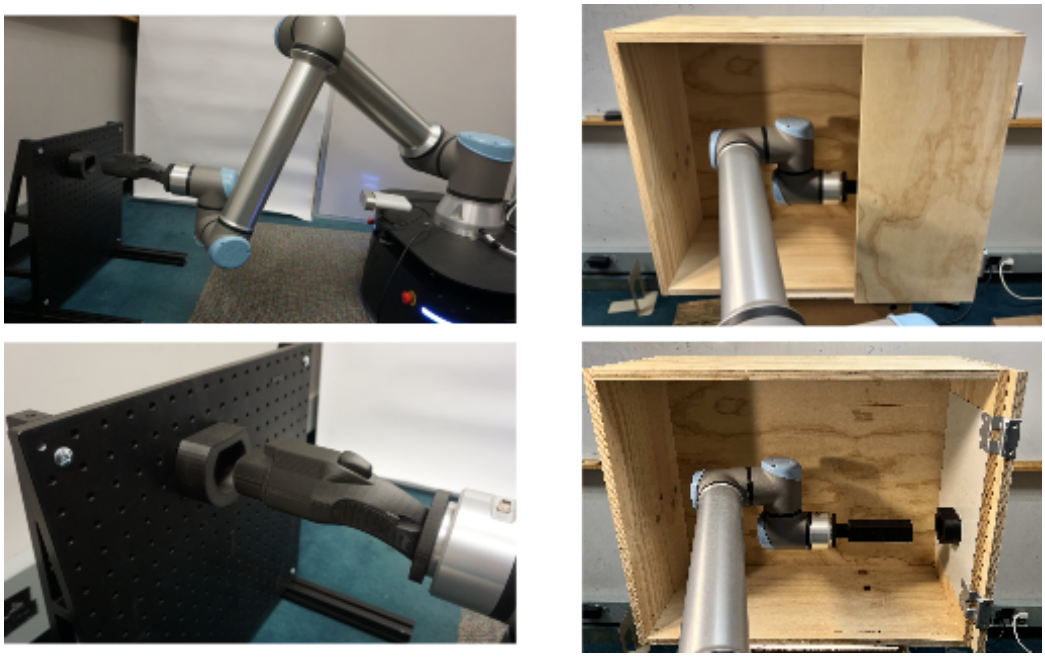

Figure 1.3: Experimental setups for plug insertion using a UR10e manipulator. Left) Port mounted on a movable base. Right) Port mounted inside a shelf and out of view of the robot.

as a Partially Observable Markov Decision Process (POMDP) [10, 11, 12]. POMDPs are a general framework for planning in partially observable environments. However, they are notoriously difficult to solve, and for most real-world problems due to the large state space and planning horizon, solving a POMDP is computationally intractable [10, 13]. However, in our context, they provide a natural mechanism to trade off between information gathering and task accomplishing.

In this thesis, we investigate planning for contact-based active information gathering aimed at localizing target objects in order to complete manipulation tasks. We develop planning frameworks that exploit problem structure and task-specific constraints to enable effectively reasoning about the underlying POMDP, allowing practical solutions that meet the performance requirements of different operational settings.

Experience-based preprocessing for semi-structured settings

In this work [14], we consider semi-structured industrial settings with stringent online performance requirements, where planners are expected to produce high-quality solutions under very tight planning time constraints. A vision system provides an initial coarse estimate of the target object pose in the form of a set of hypotheses, after which contact feedback is used to resolve the remaining uncertainty. As a result, the uncertainty addressed in this setting is relatively small, with positional errors typically on the order of a few centimeters. Due to the structured nature of the setting, the set of planning problems that may be encountered online is finite and known in advance.

Following prior work on semi-structured deterministic planning [15, 16], we propose preprocessing

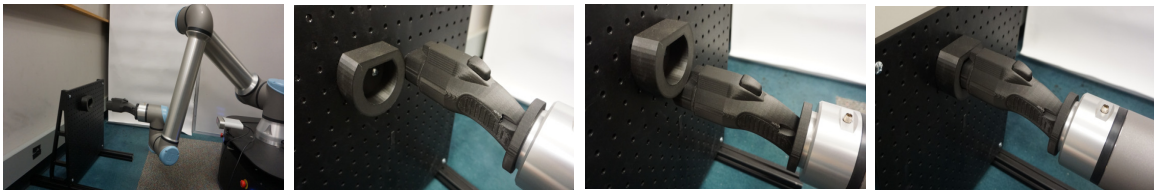

Figure 1.4: An illustration of a robot policy that contacts the port at different locations to reduce pose uncertainty sufficient to complete the plugin procedure.

a database of solution policies covering all possible problem instances. At runtime, the appropriate policy is retrieved and executed based on the current instance. Constructing such a database for POMDPs, however, is computationally challenging. To address this, we introduce a planning framework that incrementally builds the database by reusing solutions from previously solved instances. Central to this framework is Experience-based RTDP-Bel (E-RTDP-Bel), a general experience-based POMDP solver that leverages solutions from similar problems to accelerate planning while preserving strong bounds on solution quality, enabling efficient offline construction of the policy database.

Multi-modal representation for scaling up uncertainties

In [17], we address planning in significantly less structured settings where the set of possible planning problems is neither finite nor known in advance, making policy preprocessing infeasible and requiring online planning. Unlike the semi-structured case, no visual perception system is available to generate pose hypotheses. Instead, the robot relies solely on prior knowledge, resulting in substantially larger uncertainty, with positional errors on the order of hundreds of centimeters and angular uncertainty approaching 2π. An example of this setting is shown in Fig. 1.3, where the target port is fully occluded from view.

Planning under such uncertainty introduces two primary challenges. First, the scale of uncertainty renders naive particle-based belief representations (adopted in the previous setting) computationally impractical. To address this, we adopt a hierarchical belief representation that initially models uncertainty using a coarse 3D volumetric representation and transitions to a particle-based representation as uncertainty is reduced through execution. Second, computing near-optimal policies in this setting can require hours of computation, making full policy generation infeasible online. We therefore employ a greedy closed-loop planning and execution framework that repeatedly computes and executes partial policies using a heuristic-search-based anytime solver under a strict time budget. To maximize performance under limited computation, the planner focuses on likely belief outcomes and combines admissible and inadmissible heuristics to balance exploration and exploitation.

Lazy heuristic search for planning with expensive belief transitions

The planning frameworks described above rely on search-based POMDP solvers to reason about contact interactions and belief evolution. While powerful, these solvers inherently assume that belief transitions are readily available. In practice, computing belief transitions requires Bayesian updates that combine transition and observation models. In contact-rich manipulation tasks such

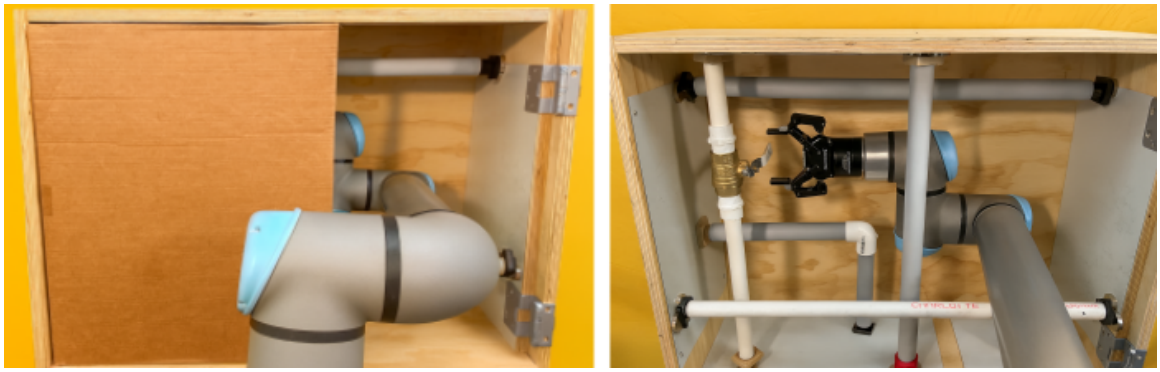

Figure 1.5: A robot reaching a shutoff valve at the back of a cluttered cabinet. As visual sensing is occluded by the shelf door (left), the robot relies on contact feedback to navigate around the obstacles (right).

as those considered in this thesis, this involves expensive geometric operations, including mesh-to-mesh collision checking. Because belief states represent distributions over system states, these costly computations must be performed for every state in the support of the belief, quickly becoming computationally prohibitive. Similar challenges arise in other robotics domains, such as outdoor navigation requiring high-fidelity physics simulation and indoor navigation involving expensive ray casting to predict sensor observations.

To address this bottleneck, we develop lazy heuristic search solvers, namely Lazy RTDP-Bel and Lazy LAO* [18], that defer the computation of expensive belief state transitions. These methods leverage Q-value estimation to identify promising actions and selectively evaluate belief transitions only when necessary, significantly reducing computational overhead. We present simple and effective techniques for constructing Q-value estimators tailored to our contact-based planning task, as well as other robotics domains with expensive belief updates. We further show that these Q-value estimates are equivalent to heuristic functions and are often readily available in practice. When the estimators are chosen to be conservative, the proposed lazy planners retain the same optimality guarantees as their non-lazy counterparts.

1.2.2 Workspace Occupancy Uncertainty

Beyond target object pose uncertainty, this thesis also addresses the problem of enabling robots to leverage contacts to operate effectively in unknown or partially known environments. Consider the task of reaching a valve located at the back of a visually occluded kitchen sink cabinet like the one highlighted in Fig. 1.5. While the robot may know that the valve is located at the rear of the cabinet, visual occlusions prevent it from observing the pipes, brackets, and other obstacles that occupy the space. Much like humans, the robot must instead rely on contact feedback to maneuver through the environment and reach the target. As it moves through the cavity, it inevitably makes contact with surrounding structures. By reasoning about where these contacts occur and combining this information with prior knowledge about typical obstacle configurations, such as pipes that often span the width of the cabinet, the robot can incrementally refine its understanding of free space and

adapt its trajectory toward the goal.

This example illustrates a broader class of manipulation problems that require contact-based reasoning in unknown environments, including retrieving objects from the back of cluttered shelves, locating switches hidden behind obstacles, or reaching through scaffolding in construction settings. In such domains, strong structural priors often exist, but the exact geometry and placement of obstacles in any given scene remain uncertain.

To address this problem, we develop a theoretically complete and empirically efficient framework for manipulation in the blind that integrates contact feedback with structural priors to enable robust operation in unknown environments [19]. The framework consists of three tightly coupled components: (i) a contact detection and localization module that uses joint torque sensing and a contact particle filter to detect and localize contacts, (ii) an occupancy estimation module that aggregates contact observations to construct a partial occupancy map and extrapolates it into unexplored regions using learned predictors, and (iii) a planning module that explicitly accounts for uncertainty in contact localization and occupancy predictions, computing collision-aware paths that efficiently complete tasks without prematurely ruling out feasible solutions. The proposed system is demonstrated both in simulation and on a real UR10e manipulator across two domestic manipulation tasks: (i) reaching a valve in a kitchen sink cabinet filled with pipes and (ii) retrieving a target object from a cluttered shelf.

1.3 Related Work

In this section, we briefly present the literature most relevant to this thesis. More detailed discussions of the related work are provided in the chapters that follow.

Task Domains: The tasks investigated in this thesis, such as high-precision insertion [20, 21, 22] and manipulation and object retrieval in cluttered environments [23, 24, 25, 26, 27], have been extensively studied in robotics. Most prior approaches that address these problems rely heavily on vision-based perception. These efforts have led to major advances in industrial automation and structured manipulation [28, 29, 30]. In this thesis, we study these same classes of manipulation problems from a fundamentally different perspective. Rather than assuming vision-based settings, we focus on partially observable and unobservable settings and investigate how contact feedback can serve as the primary sensing modality. Our goal is to enable robots to complete complex manipulation tasks in situations where vision is inadequate, by explicitly reasoning about contact interactions and the information they provide. This capability, which humans naturally employ through touch, is especially critical in unstructured and visually constrained environments.

Much of the prior work on high-precision manipulation under object pose uncertainty considers relatively small amounts of uncertainty, typically limited to narrow pose error bounds in structured environments [31, 32, 33, 34, 35]. In contrast, this thesis pushes the boundaries of uncertainty to scales that are more representative of real-world domestic settings, where visual occlusions are common and positional uncertainty can be quite high. Furthermore, the problem of manipulating in the blind in unknown workspaces, where the robot must actively reason about workspace occupancy, has

received only limited attention [36, 37, 38, 39]. Existing approaches in this space typically do not explicitly model workspace structure or exploit environmental priors, which often leads to repeated contacts with obstacles in cluttered scenes. Building on the ideas from these works, this thesis introduces a principled and scalable reasoning framework that integrates contact-driven planning with learned workspace occupancy modeling [19]. The proposed framework is evaluated in substantially more challenging settings than those previously explored, characterized by dense clutter and severe perceptual uncertainty, and demonstrates strong empirical performance. Together, these contributions advance the state of the art in manipulation in the blind and broaden the scope of contact-driven manipulation across both domains.

Tactile Sensing: Recent years have also seen significant progress in the development of rich tactile sensing technologies, including vision-based tactile sensors and high-resolution resistive sensor arrays [40, 41, 42]. While such sensors provide detailed contact information, this thesis demonstrates that complex manipulation tasks do not necessarily require sophisticated tactile hardware. Instead, we show that even high-precision manipulation can be achieved using simple and ubiquitously available contact signals. In the first part of the thesis, we rely primarily on binary collision observations, obtained reliably using wrist-mounted force torque sensors [43]. In scenarios where contact can occur anywhere along the manipulator, as in Chapter 6, we use joint current measurements to detect and localize contacts along the arm [44, 45].

Tactile Perception: A substantial body of prior work on tactile manipulation has focused on the modeling aspect of the problem. Given a history of tactile observations, these approaches seek to infer unknown parameters of the world, such as object pose, shape, or physical properties [38, 46, 47, 48, 49, 50]. In contrast, the focus of this thesis is complementary. Rather than asking only what can be inferred from tactile data, we ask how a robot should move so that it can complete its task as efficiently as possible, given that contact feedback is the primary or sole source of information. This shift places the emphasis on planning and decision-making under uncertainty, rather than on state estimation alone.

Decision-making: The central reasoning and decision-making challenge is to determine how the manipulator should move so that it can reduce uncertainty about the world through physical interactions and ultimately complete the task at hand. Much of the prior work in this area has relied on greedy decision-making strategies [47, 51, 52]. In problems involving object pose uncertainty or surface geometry uncertainty, a common approach is to select, at each step, the action that maximizes an information-theoretic objective such as entropy reduction or mutual information [51, 53, 54]. These methods are simple and often effective, but their myopic nature often leads to suboptimal behavior when successful task execution requires coordinated, long-horizon information gathering (as supported by the results presented in subsequent chapters).

From a principled standpoint, this problem is naturally a Partially Observable Markov Decision Process. POMDPs provide a rigorous framework for reasoning under uncertainty and offer a systematic way to trade off information gathering and task completion. In practice, however, POMDP

formulations are frequently avoided in robotics because general-purpose solvers are computationally expensive and often fail to meet the real-time performance constraints of manipulation systems [10]. In this thesis, we demonstrate that POMDP-based formulations of contact-driven active perception can be made practical by exploiting task structure and by developing scalable planning algorithms. Across multiple problem settings, we show that it is possible to move beyond greedy probing strategies and enable long-horizon reasoning about how contact interactions reduce uncertainty, while still achieving the efficiency required for real-world robotic deployment [14, 17, 18]. The achieved results highlight that principled decision-making under uncertainty is not only theoretically appealing but also practically viable for contact-rich manipulation tasks.

Recent years have also seen significant progress in learning-based methods for manipulation [55, 56], particularly for insertion and assembly tasks under uncertainty [57, 58, 59]. Reinforcement learning [60] and imitation learning [61] have been used to learn policies that map sensory inputs directly to control actions, often achieving impressive empirical performance in narrow task domains. Despite these successes, learned approaches typically offer limited guarantees on solution quality, robustness, and generalization. Performance can degrade substantially when tasks fall outside the training distribution, and large amounts of task-specific data are often required to achieve reliable behavior [62, 63]. In contrast, the planning-based frameworks developed in this thesis provide explicit guarantees on solution quality in the form of bounded suboptimality and completeness. By formulating contact-driven manipulation as a POMDP, we retain tight control over the decision-making process and ensure that the resulting policies systematically trade off information gathering and task execution.

At the same time, our frameworks also provide a principled backbone into which learned components can be integrated. Learned perception modules [64, 65], skill policies [66, 67, 68], and predictive models [69, 70] can be incorporated as parts of the observation and transition models or as skills/macro actions within the planning framework. In this way, the methods developed here complement learning-based approaches and provide a mechanism for combining data-driven components with strong algorithmic guarantees.

1.4 Outline

This thesis is organized into four parts.

- **Part I: Introduction**
  - **Chapter 1** motivates *manipulation in the blind* by framing contact as a central sensing modality for manipulation under uncertainty, and outlines the key problem settings and contributions explored throughout the thesis.
- **Part II: Target Object Pose Uncertainty**
  - **Chapter 2** provides background on goal-POMDPs and belief-space planning, and describes the heuristic search-based POMDP solvers that form the foundation for the methods developed in later chapters.

  - **Chapter 3** addresses the problem of high-precision manipulation under object pose uncertainty in semi-structured industrial settings by formulating contact-based active localization as a POMDP and introducing a preprocessing-based planning framework that enables fast and reliable online execution.
  - **Chapter 4** extends this line of work to less structured domestic environments with substantially larger uncertainty by introducing a hierarchical representation of uncertainty and a closed-loop, anytime planning and execution framework for online solution synthesis.
  - **Chapter 5** addresses a key computational bottleneck in these settings by introducing lazy heuristic search methods for POMDP planning with expensive belief transitions, which defer belief transition computation using fast-to-query Q-value estimators.

- **Part III: Workspace Occupancy Uncertainty**

  - **Chapter 6** studies manipulation in unknown or partially known workspaces, where uncertainty lies in the structure of the environment itself. It presents a closed-loop contact-driven framework for such settings that integrates torque-based contact detection and localization, learned occupancy predictions, and uncertainty-aware planning.

- **Part IV: Conclusion**

  - **Chapter 7** summarizes the contributions of the thesis, discusses limitations, and outlines promising directions for future research.

II

Target Object Pose Uncertainty

2 Background

Since we formulate the problem of planning under object pose uncertainty using contact feedback as a POMDP, in this chapter, we first provide a brief background on goal-POMDPs, a common formalization for minimum-cost path planning problems under stochasticity. We also describe RTDP-Bel [71] and LAO* [72], two widely used heuristic search-based POMDP solvers that form the basis for the methods developed in subsequent chapters.

2.1 Partially Observable Markov Decision Processes

POMDPs are used to model sequential decision-making problems in non-deterministic settings where the state of the system is not directly observable. Instead, indirect observations are used to infer the system state. A discrete-time goal-POMDP can be characterized by the problem tuple $P = <\mathcal{S}, \mathcal{A}, \mathcal{Z}, \mathcal{T}, \mathcal{O}, \mathcal{C}, \mathcal{G}>$.

At every timestep, the system is at an unknown state $s \in S$ and executes an action $a \in A$. This transitions the system to a new state s' dictated by the stochastic transition model $\mathcal{T}$.

$$\mathcal{T}(s, a, s') = P(s'|s, a), \ \forall \ s, s' \in \mathcal{S}, \ a \in \mathcal{A} \tag{2.1}$$

After every transition, a noisy observation $z \in \mathcal{Z}$ is made, represented by the observation model $\mathcal{O}$.

$$\mathcal{O}(s, a, z) = P(z|s, a), \ \forall z \in \mathcal{Z}, \ s \in \mathcal{S}, \ a \in \mathcal{A} \tag{2.2}$$

As the state of the system is not directly observable, the transition and observation models are used to maintain a belief of the system state at every timestep. The belief state is a probability distribution over the possible system states and is a Markovian state signal that succinctly represents the history of actions taken and observations made. After executing action a and receiving observation z at timestep $t + 1$, the belief state can be updated as follows:

$$b_{t+1}(s') = b_a^z(s') = \sum_{s \in \mathcal{S}} b_t(s) \mathcal{T}(s, a, s') \mathcal{O}(s, a, z) \tag{2.3}$$

Given a belief state and an action, the probability of transitioning to successor beliefs can be

computed by combining the transition and observation probabilities. This essentially makes the problem a completely observable MDP in the belief space. $\mathcal{C}(b, a)$ is the cost incurred for executing action a from belief state b. The shortest path problem for POMDPs can then be stated as *computing a policy π in the belief space (mapping from belief states to actions), that takes the system from a start belief b_{start} to any belief state in the goal set $\mathcal{G}$ while minimizing the expected cost incurred.*

The optimal action to take and the optimal expected cost to reach a goal state from a belief state b denoted by $\pi^*(b)$ and $V^*(b)$ respectively, are the solutions to the Bellman optimality equation for the belief MDP.

$$\begin{aligned} \pi^*(b) &= \underset{a \in \mathcal{A}}{\operatorname{argmin}} \{ \mathcal{C}(b, a) + \sum_{z \in \mathcal{Z}} P(z|b, a) V^*(b_a^z) \} \\ V^*(b) &= \min_{a \in \mathcal{A}} \{ \mathcal{C}(b, a) + \sum_{z \in \mathcal{Z}} P(z|b, a) V^*(b_a^z) \} \end{aligned} \tag{2.4}$$

2.2 Value Iteration

Value iteration [73] is a foundational algorithm in dynamic programming, extensively utilized to solve Markov Decision Processes (MDPs) and Partially Observable Markov Decision Processes (POMDPs). Value iteration identifies an optimal policy for a POMDP problem by incrementally refining a value function, until it converges to the optimal one. For goal-POMDPs value function defined over the belief states is an estimate of the expected cost to reach the goal. $V(b)$ represents the value estimate for belief state b. The optimal value function $V^*(b)$ represents the minimum cost required cost to reach the goal from b.

The value iteration algorithm for POMDPs operates by iteratively updating the value function for all possible belief states until convergence. The process begins with an initial value function $V_0(b)$, which can be initialized to zero for all b or based on a heuristic. In each iteration, known as the backup operation, the value function for each belief state b is updated using the Bellman backup operator [73] :

$$V(b) := \min_{a \in \mathcal{A}} \mathcal{C}(b, a) + \sum_{z \in \mathcal{Z}} P(z|b, a) V(b_a^z) \tag{2.5}$$

This operation is repeated until the change in the value function is below a predefined threshold, indicating convergence to the optimal value function. Upon convergence, picking the action that minimizes the value from any belief state constitutes the optimal policy π^* (Eqn. 2.4).

While value iteration can be employed to compute the optimal solution, it is computationally intensive and can not scale to problems larger than a few states. The continuous nature of the belief space makes it impractical to update the value function for every possible belief state. Additionally, the belief state represents a probability distribution over the state space, resulting in a high-dimensional space that is challenging to explore. To mitigate these challenges, various approximation techniques and heuristics have been developed. Point-Based Value Iteration (PBVI) [74] approximates the value function by focusing on a finite set of representative belief points rather than the entire belief space. Heuristic search methods like RTDP-Bel [71] and HSVI2 [75] [76] use heuristic-guided searches to efficiently explore the belief space, improving computational feasibility.

Algorithm 1 RTDP-Bel

1: **while** Not Converged **do**
2: $b = b_{start}$
3: **Sample** state s with probability $b(s)$
4: **while** $b \notin \mathcal{G}$ **do**
5: **Evaluate** the value of executing each action $a \in \mathcal{A}$ from belief state b as:

$$Q(b,a) = \mathcal{C}(b,a) + \sum_{z \in \mathcal{Z}} P(z|b,a) V(b_a^z)$$

▷ **Use $\mathbf{V(b_a^z) = heur(b_a^z)}$ if not initialized**

6: **Select** action a_{best} that minimizes $Q(b,a)$
7: **Update** value of belief state $V(b) = Q(b, a_{best})$
8: **Sample** next state s' with probability $\mathcal{T}(s, a_{best}, s')$
9: **Sample** observation z with probability $\mathcal{O}(s, a_{best}, z)$
10: **Compute** b_a^z and set $b := b_a^z$ and $s := s'$
11: **end while**
12: **end while**

2.3 RTDP-Bel

RTDP-Bel [71] is a direct adaptation of Real-Time Dynamic Programming (RTDP) [77] to partially observable domains. It is a heuristic search-based POMDP solver that performs asynchronous value iteration over the belief space, focusing computation on belief states that are reachable and relevant to the task. Rather than attempting to exhaustively compute a value function over the entire belief space, RTDP-Bel incrementally improves value estimates along simulated execution trajectories, making it well suited for large-scale problems with long horizons.

As outlined in Alg. 1, RTDP-Bel operates by performing a series of greedy rollouts. Each rollout begins from the initial belief state b_{start} (Line 2) and terminates when a belief state in the goal set $\mathcal{G}$ is reached (Line 4). At each step of a rollout, a state is sampled from the current belief, the expected outcome of executing each action is evaluated, the best action is selected, and the value estimate is updated accordingly (Lines 5–7). The next state and observation are then sampled, and the belief is updated using the observation model (Lines 8–10). Repeating this rollout procedure is guaranteed to converge to the optimal policy provided that the heuristic function used to initialize the value estimates is an admissible underestimate of the optimal value function.

The heuristic function **heur** used to initialize the value estimates (Line 5) plays a critical role in the performance of the algorithm. An informative heuristic can significantly accelerate convergence by guiding the search toward relevant regions of the belief space. As shown in [78], if the heuristic is an admissible underestimate of the optimal value function, inflating the heuristic values by a factor ε biases the search more strongly toward heuristic guidance, enabling convergence to an ε-suboptimal solution much more rapidly.

Algorithm 2 LAO*

1: $G^* \leftarrow \{b_0\}$
2: **while** G^* has non-terminal tip states **do**
3: **Identify** a non-terminal tip state b in G^*
4: **Evaluate** each action $a \in \mathcal{A}$ from b
5: **Compute** value of each action as:

$$Q(b,a) = c(b,a) + \sum_{z \in \mathcal{Z}} P(z|b,a)V(b_a^z)$$

▷ $V(b_a^z) = \text{heur}(b_a^z)$ if uninitialized

6: **Select** action a_{best} that minimizes $Q(b,a)$
7: **Update** value $V(b) = Q(b, a_{best})$
8: **Create** set Z with b and ancestors of b in G^*
9: IMPROVEVALUES(Z) ▷ Value iteration
10: **Update** G^* to current best solution
11: **end while**
12: IMPROVEVALUES(G^*) ▷ Value iteration
13: **if** $G^* \neq$ current best solution **then**
14: **Update** G^* to current best solution
15: **Goto** Line 2
16: **end if**
17: **return** G^*

2.4 LAO*

LAO* [72], outlined in Alg. 2, is another widely used heuristic search-based solver for POMDPs that, like RTDP-Bel, focuses computation on belief states that are relevant to the task. Rather than performing rollouts, LAO* explicitly constructs and refines a partial solution graph over the belief space. This allows the algorithm to interleave forward expansion with dynamic programming backups while ensuring that belief states unreachable from the start belief b_0 under the current best policy are never explored.

LAO* maintains a partial solution graph G^* consisting of all belief states reachable from b_0 under the current policy. The algorithm iteratively expands and updates G^* until no non-terminal tip states remain. A belief state b is classified as a non-terminal tip state if it is neither a goal belief nor has been expanded, meaning that its successor beliefs b_a^z have not yet been generated.

When such a state is encountered, all possible actions from b are evaluated and their corresponding successor beliefs are computed. The optimal action is selected based on the current Q-value estimates, and the value of b is updated accordingly (Lines 4–7). Next, a set Z is constructed containing b and its ancestor beliefs in G^*, and value iteration is performed over Z to update both value estimates and optimal actions (Line 9). The solution graph G^* is then updated to reflect these changes (Line 10). This process repeats until no non-terminal tip states remain in G^*.

Once all reachable belief states have been expanded, value iteration is performed over the entire solution graph. If these updates alter the current best policy, G^* is revised, and the algorithm returns to the expansion phase[1]. If the value function converges, the algorithm terminates and returns the

[1]This description simplifies the original presentation of LAO for clarity, while preserving its fundamental properties.

final solution policy represented by G^* (Lines 12–17). As with RTDP-Bel, the heuristic function plays a central role in guiding the search toward promising regions of the belief space. When the heuristic is admissible, LAO* converges to the optimal solution, and when the heuristic is inflated, bounded suboptimality guarantees apply.

3 Experience-based preprocessing for semi-structured settings

3.1 Introduction

Manipulation tasks like plug insertion (Fig. 3.1) or assembly have low tolerance to pose estimation errors. Errors as low as 2mm can cause task failure[1]. However, it is infeasible for perception to always perfectly localize the object of interest (which for plug insertion is the port for the plug). This is especially the case for semi-structured environments where the variations in working conditions like lighting changes, occlusions, and changes in the relative pose between the camera and the object, make perception challenging.

A common strategy to counteract this problem is to use visual servoing [79, 80]. Visual servoing is a technique where a robot is controlled using visual feedback from a sensor (typically) mounted on the robot's end effector. A control loop determines the movement of the robot's end effector such that an error function based on the visual input is minimized. However, these approaches are subject to challenges all perception systems are affected by, including varying lighting conditions and occlusions (created by the object in-hand or by obstacles in the environment) [81]. In this work, we explore an alternate avenue by allowing the robot to use its contact/touch modality to accurately localize the object of interest and complete the high-precision insertion tasks, thereby circumventing the above-mentioned challenges.

Instead of requiring a single perfect pose estimate, we relax the perception system to estimate a set of hypothesis poses within which the object of interest lies. The manipulator then utilizes the occurrence of contacts (or the lack thereof) during execution as observations to exactly localize the object and thereby complete the task. We demonstrate that this binary contact observation is extremely powerful and can reliably localize the object at a resolution fine enough to complete high-precision insertion tasks. Conceptually, we frame this as a planning problem under pose uncertainty where the robot observes contacts during execution.

In this work, we focus on insertion tasks in semi-structured industrial settings where the set of initial pose distributions is finite (which in turn implies a finite set of possible planning problems). Prior works which have looked into semi-structured domains with finite problems have proposed enu-

[1] Tolerance numbers obtained from experimental data

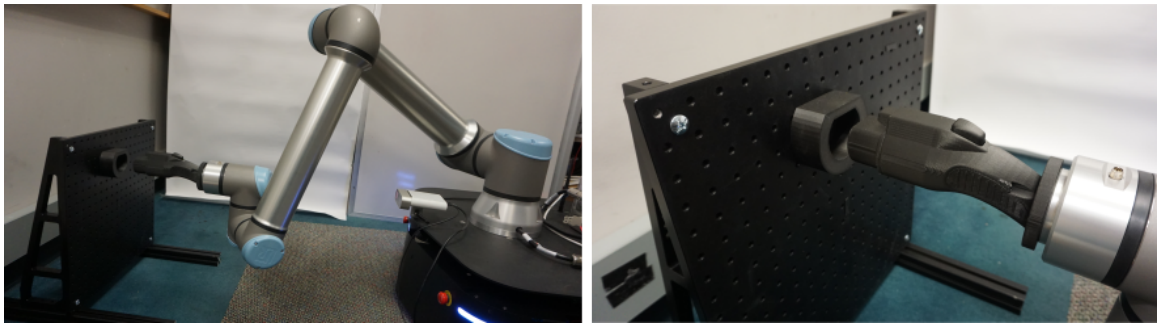

Figure 3.1: Experimental setup for plug insertion using a UR10e manipulator.

merating the set of possible problems that can be encountered online and preprocessing a database of solutions [15, 16]. However, prior works solve deterministic problems, which only require the database to contain solution *paths*. Due to pose uncertainty, our database needs to contain solutions for Partially Observable Markov Decision Processes (POMDPs) i.e. *policies*. Naively constructing a database of such solutions is computationally infeasible due to the curses of dimensionality and history associated with solving POMDPs. Hence, it is imperative that intelligent choices are adopted to make the process of database creation scalable.

We exploit the fact that the problems in the database are similar to each other and propose a planning framework that iteratively constructs the database by using the solution of one problem as experience for solving the next. We propose a general experience-based POMDP solver, Experience-based RTDP-Bel (E-RTDP-Bel), that effectively utilizes prior experiences (solutions from similar planning problems) to speed up planning queries while maintaining strong bounds on solution quality and use it in our framework to efficiently construct the database of solution policies.

The performance of the overall framework is demonstrated in the real world on the task of inserting a plug into a port using a UR10e manipulator. We utilize an ICP-based registration framework [82] to identify a discrete distribution of port poses. The results from the experiments highlight the robustness of the framework, succeeding 95% of the runs. We also demonstrate the generality of the framework by using it on the task of pipe assembly in simulation. Performance analyses presented in Section 3.5 show that the proposed E-RTDP-Bel algorithm improves planning times by over a factor of 100 while maintaining strong solution quality, making the database construction process feasible.

Succinctly, our contributions are:

1. Developing a *preprocessing-based planning framework* that solves high-precision insertion tasks under pose uncertainty for semi-structured settings.

2. Developing a general *experience-based RTDP-Bel algorithm* that uses experience from similar problems to significantly speed up planning queries while maintaining strong bounds on solution quality.

3. Demonstrating the effectiveness of using binary contact information for accurate object localization.

3.2 Related work

We first discuss different frameworks used for solving high-precision insertion tasks. Then, we delve into works that are closely related to our algorithmic contributions.

3.2.1 Framework

Visual servoing is a popular class of approaches used for solving high-precision insertion. Classical approaches in this field have explored three major ideas i) **Image-based visual servoing** which defines error metrics in the image space and computes end effector controls that minimizes the error [83], ii) **Position-based visual servoing** which continually estimates the pose of the object (using its model) and moves the end-effector towards a target pose [20], and iii) **Hybrid** approaches which intelligently combine the two ideas [84]. Recent works have also trained deep neural network policies that map images to motor controls [85]. In this work, we instead explore the use of an alternate sensing modality, tactile sensing, to close the feedback loop.

Many prior works which incorporate tactile feedback focus on developing Particle Filter and Bayesian estimation methods to compute object poses that best explain the sequence of tactile observations made [38, 46, 47]. However, our work focuses more on active localization using tactile feedback, i.e. reasoning about the sequence of actions to take to quickly localize the object of interest. A popular idea here is to utilize a myopic framework that interleaves planning and execution [47, 51]. In each planning cycle, actions are sampled and the action that maximizes an information gain metric is executed by the robot. This planning and execution cycle is repeated until the uncertainty is reduced sufficiently. As a consequence of being myopic and only reasoning about the next best action to take (as opposed to the sequence of actions to complete the task), these algorithms sacrifice solution quality which is critical in industrial settings. On the other hand, by formulating and solving the planning problem as a POMDP, we are able to achieve significantly better solution quality. This is evident from the performance comparisons with [51] presented in Section 3.5.

Recently, reinforcement learning-based approaches for peg insertion have gained popularity [86]. [87] uses a long short-term memory model (LSTM) to estimate the Q value function to achieve peg-in-hole assembly. [88] proposes a model-driven DDPG algorithm to learn the general assembly policy for multiple peg-in-hole problems. We differentiate ourselves from prior literature in this field by posing the problem as an active localization task and explicitly reason about using contacts to localize the port/hole.

3.2.2 Algorithm

On an algorithmic level, we are inspired by [89] which introduces the notion of Experience Graphs for deterministic settings. Their approach uses solutions from similar episodes to construct a graph that represents the underlying connectivity of the space required to complete the task. Given a new planning query, the algorithm attempts to reuse this graph as much as possible by constructing a heuristic function that makes the search prioritize exploring regions around prior experiences. In this work, we extend the idea of Experience Graphs to POMDPs. A variety of algorithms have

been proposed to solve POMDPs. RTDP-Bel [71] is a popular heuristic-search-based algorithm that exhibits great sample efficiency and provides strong solution guarantees. We augment RTDP-Bel to reuse prior experiences to significantly speed up planning while continuing to maintain solution guarantees.

The idea of preprocessing a database of solutions to eliminate or speed up planning online has existed in different forms over the years [15, 90, 91]. Due to the computational complexity of solving POMDPs and the need for strong online performance, our approach utilizes a similar idea.

3.3 Problem Formulation

In this section, we formulate the active localization using contacts problem as a POMDP. Given a robot manipulator, let $\mathcal{R}$ represent its state space and $\mathcal{A}$ the discrete action space. The observation space $\mathcal{Z}$ comprises

1. The robot's state (encoder readings), and
2. A binary flag indicating the occurrence of contacts/collisions (F/T sensor readings or joint torques)

Let H_{start} represent the discrete distribution of hypothesis poses of the object of interest identified by a perception system. For ease of explanation and discussion, we assume this distribution is uniform (our approach can be extended to non-uniform distributions with minor modifications). This allows us to represent H_{start} as a discrete set of hypothesis poses $H_{start} = \{h_1, h_2, ... \, h_n : h_i \in SE(3)\}$.

Unlike typical robotics problems, in our case, the stochasticity in transitions and observations arises as a result of uncertainty in the model of the environment i.e. the object pose. We assume the dynamics and the observations to be perfect given an object pose h_i. Meaning, $P(r'|r, a, h_i)$ and $P(z|r, a, h_i)$ are all either zero or one. The prevalence of high-quality manipulators in industries and the very simple and reliable observation space (contacts) make the perfect dynamics and observations assumption practical. Our real-world results detailed in Section 3.5 are a testament to this. We also assume that the environment is static, i.e., robot actions do not change the state of the object being localized.

Hence, the state of the system s for our problem contains both the robot state and the environment state, i.e. the object pose. This is represented by the tuple (r, h). If $s = (r, h)$ and $s' = (r', h')$, the transition and observation models for our system state can be defined as follows.

$$\mathcal{T}(s, a, s') = P(s'|s, a) = \begin{cases} 1 & h' = h, \;\; P(r'|r, a, h) = 1 \\ 0 & \text{Otherwise} \end{cases}$$
$$\mathcal{O}(s, a, z) = P(z|s, a) = \begin{cases} 1 & P(z|r, a, h) = 1 \\ 0 & \text{Otherwise} \end{cases} \tag{3.1}$$

Essentially the transition model describes that executing an action a from a fully observed state $s = (r, h)$ will deterministically transition the robot to a new state r' while leaving the environment

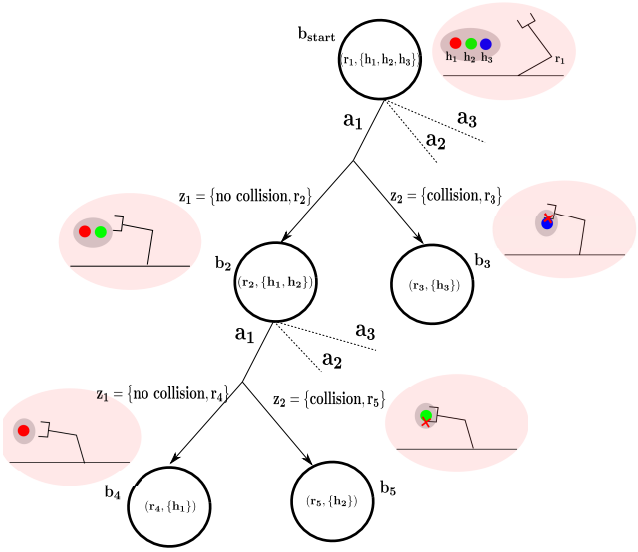

Figure 3.2: Belief tree for the active localization using contacts problem. The robot uses the occurrence of contacts (or the lack thereof) to reduce uncertainty, i.e., the hypothesis pose set.

state (i.e. the object pose) unchanged.

The belief state is a probability distribution over the state space. Under the uniform initial pose distribution and perfect observation assumptions, the belief state can be represented by a tuple (r, H), where r is the robot state and $H = \{h_1, h_2..\}$ is the discrete set of hypothesis poses. As the task is to localize the object, the set of goal beliefs $\mathcal{G}$ is given by belief states that contain exactly one hypothesis pose.

Fig. 3.2 represents a portion of the belief tree. Executing an action from a belief state that results in different observations under different hypotheses leads to successor beliefs with reduced uncertainty. As can be seen in the figure, executing action a_1 from belief state b_2 which contains two hypothesis poses $\{h_1, h_2\}$, results in a collision observation under h_2 and uninterrupted execution under h_1. Executing this action allows us to disambiguate between the hypotheses h_1 and h_2 thereby localizing the object.

The planning problem representing the active localization task can therefore be stated as *given a robot start state r_{start} and the set of initial hypothesis poses H_{start}, compute a belief space policy that reduces the hypothesis set to a single element while minimizing the expected distance traveled.*

The assumptions we make about the above-formulated active localization problem can be concisely listed as follows:

A1 Realizable setting, i.e., the groundtruth object pose is contained in the hypothesis set H_{start}.

A2 Deterministic transitions and perfect observations given the environment state, i.e., object pose.

A3 Known geometric model of the object of interest (used to compute expected observations).

A4 Static environment, i.e., the object of interest remains static upon interaction.

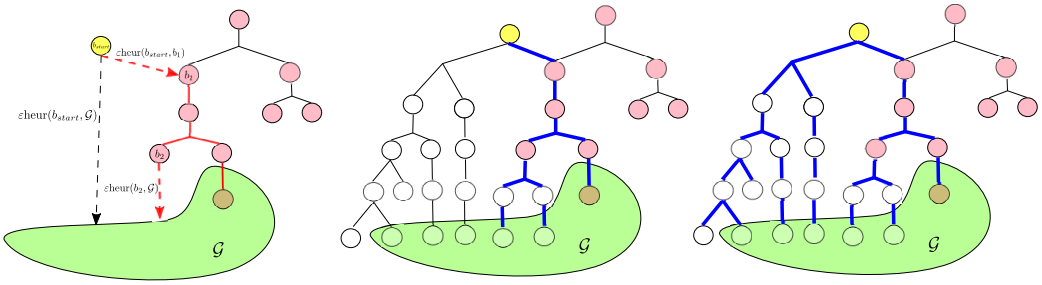

Figure 3.3: left) b_{start}, $\mathcal{B}^E$, and $\mathcal{G}$ are highlighted in yellow, maroon, and green respectively. The instantaneous transitions/jumps relevant for computing heur^E for b_{start} are indicated with dashed arrows. Under a sufficiently high ε, the experience heuristic prefers the policy highlighted in red, encouraging the search to move towards $\mathcal{B}^E$, and the value of this policy is used as heur$^E(b_{start})$. middle and right) Contrast the region of the belief space $\mathcal{B}$ explored by RTDP-Bel to converge to a solution (highlighted in blue) when using heur^E (under high ε) vs heur, respectively.

It should be emphasized that our approach can be extended to incorporate noisy transition and observation models with minor adjustments. However, in such cases, we can anticipate a decrease in planning speed due to the expanded search space. The assumption of a realizable setting ensures that the observations obtained from real-world interactions align with at least one of the hypotheses proposed in H_{start}. If none of the hypotheses can explain the observations, all of them will be eliminated, indicating that the task is unachievable.

3.4 Approach

We are interested in semi-structured domains where the set of localization planning problems that can be encountered is finite and known ahead of time. This is defined by a fixed robot start state r_{start} and the finite set of initial distributions that can be encountered $\mathcal{H} = \{H^1_{start}, H^2_{start}...\}$. As the set of problems is known, we propose to preprocess solutions to the problems ahead of time and construct a database of policies $\Pi = \{\pi_1, \pi_2...\}$ that localize the object of interest for each possible distribution H^i_{start}. Online, based on the problem at hand, the solution policy from the database is retrieved and executed.

Constructing this database naively by solving each problem independently is computationally expensive. Based on the key observation that the problems in the database are similar to each other, we propose to iteratively construct the database by using the solution of one problem as experience for solving the next. To this end, we develop a general experience-based POMDP solver, E-RTDP-Bel, that uses solutions of similar problems to speed up planning queries and use it in our problem domain.

We start off by describing RTDP-Bel [71], the POMDP solver that forms the basis of our planning framework. Followed by which we describe E-RTDP-Bel and move on to describing how it is used in the context of our domain.

3.4.1 E-RTDP-Bel: Experience-Based RTDP-Bel

Idea

The intuitive idea behind Experience-Based RTDP-Bel is that *similar planning problems likely have similar solutions*. Hence, given a problem and the solution to a similar problem, exploring regions of the belief space around the solution is likely to solve the current problem.

Different from the original belief MDP of the planning problem $\mathcal{B}$, the approach explicitly maintains an experience MDP $\mathcal{B}^E$ which is a significantly smaller subset of the original belief space, i.e. $\mathcal{B}^E \subset \mathcal{B}$. $\mathcal{B}^E$ is constructed from solutions of previous planning problems and at its essence represents the portion of the belief space relevant to the planning problems. A simple technique to construct $\mathcal{B}^E$ would be to take the union of all transitions in previous solution policies. The key idea behind our approach then is to speed up planning queries by reusing portions of solution policies from previous problems as much as possible, i.e., portions of $\mathcal{B}^E$. This reduces the need for searching large areas of the original belief space $\mathcal{B}$.

We achieve this by constructing an intelligent heuristic function, the experience heuristic (heur^E), using the original heuristic defined for the problem domain (heur), the goal set for the problem $\mathcal{G}$, and the experience MDP $\mathcal{B}^E$. The experience heuristic biases the search iterations in RTDP-Bel towards $\mathcal{B}^E$ when following parts of $\mathcal{B}^E$ is likely to get the search closer to the goal. We use $\text{heur}^E(b)$ as shorthand for $\text{heur}^E(b, \mathcal{G})$ and is the heuristic estimate of the expected cost to reach the goal set of the problem from the belief b,

$$\text{heur}^E(b) = \min \begin{cases} \varepsilon \text{heur}(b, b') + \text{heur}^E(b') & \forall\ b' \in \mathcal{B} \\ \mathcal{C}(b, a)\ + \sum_{z \in \mathcal{Z}} P(z|a, b)\ \text{heur}^E(b_a^z) & \forall\ (b, a) \in \mathcal{B}^E \end{cases} \tag{3.2}$$

Essentially, given a belief state b, the experience heuristic computes the value of the optimal policy from b to $\mathcal{G}$ that is composed of two kinds of transitions, i) Instantaneous jumps that take the system between any two belief states in the belief space $\mathcal{B}$. The cost of these instantaneous transitions is equal to the heuristic value inflated by the penalty term ε. ii) Transitions that are part of the belief MDP $\mathcal{B}^E$ which incur their true cost. By penalizing the (pseudo) transitions that lie outside the experience MDP, the heuristic encourages the search to reuse as much of $\mathcal{B}^E$ as possible. As the penalty ε increases, the search will go farther out of its way to reuse experiences. We run RTDP-Bel with this newly defined experience heuristic to speed up planning. We refer to this approach as Experience-based RTDP-Bel (E-RTDP-Bel). Fig. 3.3. provides a visual depiction of the experience heuristic idea.

Implementation

Computing the experience heuristic as defined in Eqn. 3.2. is computationally intractable. Given a belief state b, computing heur^E involves reasoning over the instantaneous jumps from b to every state in $\mathcal{B}$. However, if the original heuristic follows the h-consistency property, i.e., $\text{heur}(b_1, b_3) \leq \text{heur}(b_1, b_2) + \text{heur}(b_2, b_3), \forall\ b_1, b_2, b_3 \in \mathcal{B}$, then $\varepsilon\text{heur}(b) \leq \varepsilon\text{heur}(b, b') + \varepsilon\text{heur}(b')\ \forall\ b' \notin \mathcal{B}^E \cup \mathcal{G}$.

Algorithm 3 EXPERIENCE HEURISTIC

```
1: procedure PRECOMPUTE VALUES(B^E, G, heur)
2:     for b ∈ B^E do
3:         heur^E(b) = ε heur(b)                          ▷ initialization
4:     end for
5:     while NOT CONVERGED do
6:         for b ∈ B^E do
7:             Compute heur^E(b) as defined in Eqn. 3.3.
8:         end for
9:     end while
10: end procedure
```

Therefore, for all $b \in \mathcal{B}$, the instantaneous actions can be restricted to a significantly smaller subset, $b' \in \mathcal{B}^E \cup \mathcal{G}$. Now, instead of reasoning over instantaneous transitions to every state in the belief space, we only need to reason about jumps to the experience MDP $\mathcal{B}^E$ or the goal set $\mathcal{G}$.

$$\text{heur}^E(b) = \min \begin{cases} \varepsilon\text{heur}(b, b') + \text{heur}^E(b') & \forall\ b' \in \mathcal{B}^E \cup \mathcal{G} \\ \mathcal{C}(b, a)\ + \sum_{z \in \mathcal{Z}} P(z|a, b)\ \text{heur}^E(b_a^z) & \forall\ (b, a) \in \mathcal{B}^E \end{cases} \tag{3.3}$$

Computing the experience heuristic for every state encountered in the search is still non-trivial. However, computing heur^E for any state b encountered in RTDP-Bel can be reduced to a simple linear pass over the states in $\mathcal{B}^E \cup \mathcal{G}$ if the experience heuristic for the states in $\mathcal{B}^E$ is known.

$$\text{heur}^E(b) = \min_{b' \in \mathcal{B}^E \cup \mathcal{G}} \varepsilon\text{heur}(b, b') + \text{heur}^E(b') \qquad \forall\ b \notin \mathcal{B}^E \cup \mathcal{G} \tag{3.4}$$

As the experience heuristic for states in $\mathcal{B}^E$ is only dependent on $\mathcal{G}$ and $\mathcal{B}^E$ itself, prior to running RTDP-Bel we precompute heur^E for all states in $\mathcal{B}^E$ by performing a series of asynchronous Bellman updates (outlined in Alg 2).

For any belief state b, $\text{heur}^E(b)$ is upper bounded by $\varepsilon\text{heur}(b)$ (as $\text{heur}^E(b') = 0$, if $b' \in \mathcal{G}$). Hence, if heur is admissible, i.e., underestimates the cost of transitions, the solution is guaranteed to be ε-suboptimal. This property directly follows from [78], which proved that a solution policy found by RTDP-Bel using an ε-inflated admissible heuristic is guaranteed to be ε-suboptimal.

3.4.2 Using E-RTDP-Bel For Database Preprocessing

In this subsection, we show how E-RTDP-Bel is used to speed up constructing the database of policies Π (Alg. 4).

Given the set of localization problems defined by the robot start state r_{start} and the set of initial object pose distributions $\mathcal{H} = \{H^1_{start}, H^2_{start} ...\}$, we can maximize the benefit of E-RTDP-Bel by solving the problems in an intelligent order. The closer the experience MDP $\mathcal{B}^E$ is to the solution policy of the problem being solved, the more speed up we are likely to observe, as larger portions of $\mathcal{B}^E$ can be reused. Hence, we choose to solve the problems in increasing order of uncertainty as

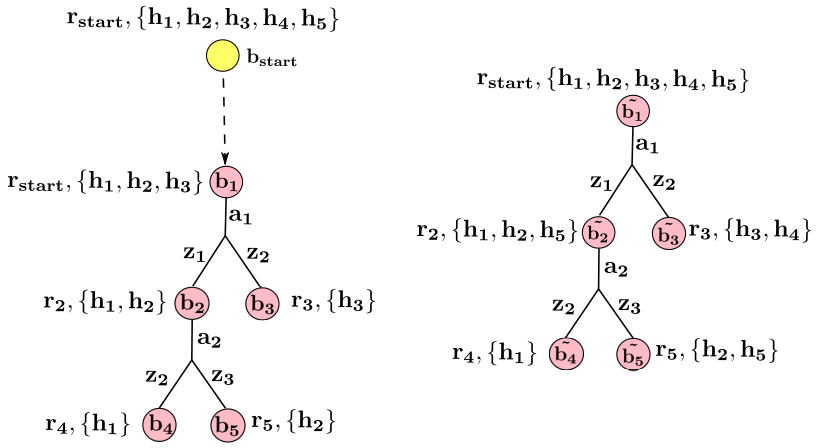

Figure 3.4: left) Getting from the start belief of a more complex problem p' (yellow) to the solution of a simpler problem p, i.e., $\mathcal{B}^E_{naive}$ (maroon) can be non-trivial. right) Rolling out the solution of the p from p'_{start} makes the experience directly accessible from p'_{start}.

it will allow us to reuse solution policies from simpler problems to construct rich $\mathcal{B}^E$'s which more closely resemble solutions to complex problems.

A subtle but important observation is that the start beliefs between the problems can be very different due to the initial uncertainty H^i_{start} being different. For example, let the start belief of a simple problem p be $p_{start} = (r_{start}, \{h_1, h_2, h_3\})$ and the start belief of a more complex problem p' be $p'_{start} = (r_{start}, \{h_1, h_2, h_3, h_4, h_5\})$. In this case, directly using the solution policy of p as the experience MDP for solving p' might not be beneficial as getting from the start state of p' to the experience is non-trivial in the belief space. We refer to the experience MDP which directly uses the solution of a similar problem as $\mathcal{B}^E_{naive}$. Hence, in this case getting from p'_{start} to $\mathcal{B}^E_{naive}$ involves eliminating at least 2 hypothesis poses ($\{h_4, h_5\}$), and identifying actions that can achieve this is non-trivial. As a result, the impact of the experience is reduced.

To counter this issue, we transform $\mathcal{B}^E_{naive}$ in the belief space so it is easily accessible from p'_{start}. We achieve this by rolling out the policy of p from p'_{start} (as opposed to p_{start} itself) and using the resulting MDP as the experience MDP $\mathcal{B}^E$. To roll out the solution policy computed for p from p'_{start}, we represent the solution policy as a mapping from histories to actions (as opposed to belief states to actions). Here, history corresponds to the sequence of actions taken and observations made. For example, in Fig. 3.4, the belief state b_2 can be abstracted out as the state reached by executing action a_1 and receiving observation z_1. In the modified representation of the policy, history $\{a_1, z_1\}$ is mapped to action a_2. Hence, action a_2 is applied from the history $\{a_1, z_1\}$ in the more complex problem as well (which corresponds to belief state $\tilde{b_2}$). Constructing $\mathcal{B}^E$ through this rollout procedure ensures that the experience can be reused directly, maximizing the speedup. This is evident from the ablation studies presented in Section 3.5.

Algorithm 4 PREPROCESSING FRAMEWORK

procedure PREPROCESS($r_{start}, \mathcal{H}$)
  $\Pi = \emptyset$
  Order $\mathcal{H}$ in increasing order of uncertainty
  for $H^i_{start} \in \mathcal{H}$ **do**
    $b_{start} = (r_{start}, H^i_{start})$
    Pick π^j from Π with most similar start state uncertainty
    $\mathcal{B}^E$ = Rollout π^j from b_{start}
    Use Alg. 3 to precompute $\text{heur}^E(b)\ \forall\ b \in \mathcal{B}^E$
    Use Eqn. 3.4. to compute $\text{heur}^E(b)\ \forall\ b \notin \mathcal{B}^E \cup \mathcal{G}^i$
    Compute solution policy π^i from b_{start} to $\mathcal{G}^i$ using RTDP-Bel with heur^E ▷ E-RTDP-Bel
    Π.INSERT(π^i)
  end for
end procedure

3.5 Results

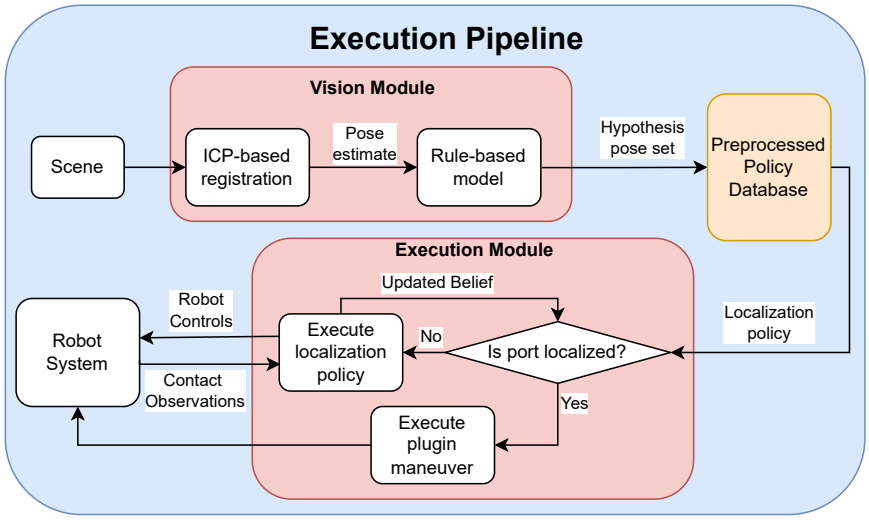

Figure 3.5: Schematic diagram of the overall system. The preprocessed policy database is constructed using E-RTDP-Bel as outlined in Section 3.4.2.

We evaluate the performance of our framework on i) A real-world plug insertion task incorporating 3D positional uncertainty of the plug port, and ii) A simulated pipe assembly task incorporating 4D pose uncertainty of the pipes.

3.5.1 Plug Insertion Task

Setup

The task is to insert a plug into its port using a UR10e manipulator. The position of the port is constrained to be within a bounded volume (2m x 2m x 2m) and is allowed to yaw within a

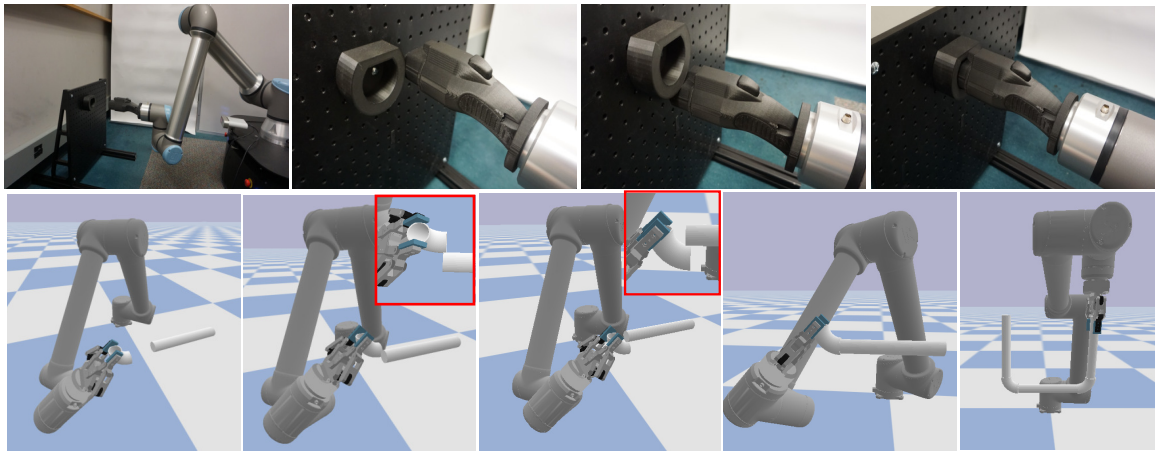

Figure 3.6: A run of the proposed framework on the plug insertion task in the real world (top row) and the pipe assembly task in simulation (bottom row). We observe the robot making contacts at different locations to localize the object of interest and completing the insertion task upon localization. In the pipe assembly case, the construction of complex structures can be modeled as a sequence of insertions.

range of -15 to +15 degrees (roll, pitch are fixed). As 2mm errors in position estimation can cause failures, the task is modeled under 3D positional uncertainty. The action set $\mathcal{A}$ is composed of unit movements in the end effector space. Though a similar discrete set could be defined in the joint space, movements in the end effector space were simpler and provided a more informative and natural set that was better suited for the task at hand (gaining information quickly).

We use a simple perception module rooted in ICP (Iterative Closest Point) [82] to identify a discrete distribution of hypotheses port poses H_i. Given a scene, we capture a point cloud from the static camera (Microsoft Azure Kinect DK) and use ICP (with the model of the port) to estimate the port pose. This pose estimate is converted into a discrete distribution using a simple rule-based approach. We represent the 3D positional uncertainty as a cuboid centered around the pose estimate. This is then discretized at a resolution of 2mm to create the hypotheses set H_i. The rules for generating the cuboid of uncertainty given the pose estimate are constructed by comparing the groundtruth poses with the ICP (noisy) estimates at different regions in the workspace. Given this rule-based uncertainty model and the workspace, we query the model at all (discretized) poses within the workspace to create the set of possible initial distributions $\mathcal{H} = \{H_1, H_2 \dots\}$. The set contains initial uncertainties ranging from 4mm to 3cm along each axis, meaning that H_i contains up to 15^3 poses. The uncertainty model is not the focus of our framework, more sophisticated approaches can be used with our planning framework. A schematic diagram of the entire system and the different modules involved can be found in Fig. 3.5.

Impact of Reusing Experiences

Table 3.1 presents results that demonstrates the impact of the developed E-RTDP-Bel algorithm in constructing the database of policies Π. Each algorithm is given a timeout of 500 seconds for

Table 3.1: Statistics For Database Construction (Plug Insertion)

Metrics	**Planning Algorithm**						
	RTDP-Bel				E-RTDP-Bel		
	$\varepsilon = 1$	$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$	$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$
Success Rate (%)	78.9	81.1	84.2	86.5	**100**	**100**	**100**
Relative Speedup	1.0	1.22	1.56	2.32	113.24	283.13	**551.97**
Relative Expected Cost	**1.0**	1.08	1.78	2.19	1.12	1.83	2.46

each planning problem. If the algorithm fails to solve the problem within the timeout, a failure is recorded. The speedups and costs are presented relative to RTDP-Bel (run with a suboptimality bound of $\varepsilon = 1$). From the table, it is clear that E-RTDP-Bel dominates RTDP-Bel providing average speedups of over a factor of 100 while maintaining similar costs. The table also presents the tradeoff between solution quality and planning time dictated by the penalty ε. As ε increases, E-RTDP-Bel encourages the search to go farther out of its way and reuse larger portions of previous solutions thereby converging quicker at the expense of solution quality. The performance of RTDP-Bel under different suboptimality bounds highlights the relative impact of E-RTDP-Bel as the suboptimality bound increases.

Performance of the Overall Framework

Table 3.2: Performance of the Framework on plugin and assembly tasks

Metrics	**Task**	
	Plug Insertion (Real World)	Pipe Assembly (Simulation)
Success Rate (%)	95	100
Execution Time (s)	22.3	16.9

Table 3.2 presents the performance of our framework averaged over 50 runs in the real world. In each run, the port is moved to a random pose within its workspace. We observe a strong success rate of 95%. The source of the 5 failures is incorrect perception, wherein, the groundtruth pose is not contained within the hypothesis set identified by perception.

We also evaluate a popular online approach, Touch Based Localization (TBL) [51], on our problem domain. TBL interleaves planning and execution. In each iteration of planning, multiple actions are

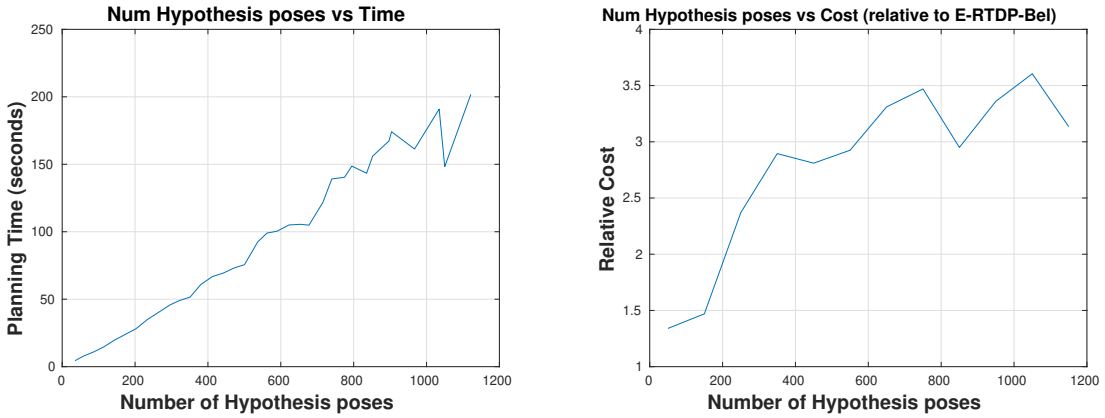

Figure 3.7: Performance of TBL on the task of plug insertion. left) Initial uncertainty (number of hypothesis poses) vs planning time. right) Initial uncertainty (number of hypothesis poses) vs Cost (relative to RTDP-Bel).

sampled, and based on the current belief their expected information gain is computed. The action that maximizes the information gain metric is chosen for execution. Upon execution, the observation received is used to update the belief, and the process is repeated. This approach is inefficient in our domain for two reasons. The first is the computational expense associated with it, especially under high uncertainties. Evaluating information gain of an action or even maintaining the belief distribution, involves forward simulating the action under all hypothesis poses and computing the expected observation. This process involves expensive 3D mesh collision checks. In our problem, hypothesis sets can contain thousands of poses resulting in several thousand collision checks for each action. Although TBL is faster than solving the problem as a POMDP, it is not fast enough to be used online. The second reason is that TBL is myopic. It only reasons about the next best action to execute and not the sequence of actions to execute to complete the task, thereby compromising solution quality. This effect is exacerbated under high uncertainties. Fig. 3.7. presents how the planning times and solution qualities scale with uncertainty. For our evaluation, we sampled 100 actions in the vicinity of the hypothesis poses in each planning iteration.

A simple Image-based visual servoing approach was also evaluated. A RealSense D435i camera was mounted on the wrist of the manipulator and provided a continuous video stream of the region of interest. SIFT [92] was used for registering the features between the current and the target image. The positional errors between the features in the image space (from the current to the target image) were used to compute the robot controls. The lack of clear contrast between the object of interest and its background (black colored port mounted on a black background) combined with the fact that the plug in hand occluded most of the port (especially in close proximity) led to poor feature registration and as a result only 1 success out of 10 runs. To compare against the performance of the approach under ideal conditions, a clearly visible red tape that could be reliably identified in the images (including those from close proximity) was attached to the port. Instead of using the SIFT features, we used the corners of the tape (which could be reliably located using a simple color mask) as features for the IBVS algorithm. The performance was significantly better under such ideal conditions succeeding in 9 out of 10 runs (with an average execution time of 11 seconds). These

Table 3.3: Preprocessing Framework Ablation Study

Metrics	Planning Framework		
	OURS	NAIVE	RANDOM
Success Rate (%)	**100**	**100**	95.5
Relative Speedup	**1.0**	0.23	0.11
Relative Expected Cost	1.0	**0.95**	0.98

results reiterate the challenges faced by vision-based algorithms and highlight the need for using tactile feedback.

We highlight the impact of the choices we make in our preprocessing-based planning framework through ablations presented in Table 3.3. Here, NAIVE uses $\mathcal{B}^E_{naive}$ as the experience MDP instead of the proposed rollout procedure to construct $\mathcal{B}^E$ (Section 3.4.2). RANDOM orders the problems randomly to highlight the benefit of ordering problems in increasing order of uncertainty. All variants run E-RTDP-Bel with $\varepsilon = 2$, and their performance in constructing the database of solutions is presented relative to our framework. We observe that NAIVE and RANDOM are 5 and 10 times slower than our framework respectively, highlighting the impact of the *quality* of experience on planning performance. By ordering the problems in terms of increasing uncertainty and by performing the rollout procedure, we are able to construct Experience MDPs that are more relevant (larger portions of the experience can be reused) and easily accessible, resulting in stronger performance.

Overall, our framework provides a strong combination of real-world robustness, planning times, and solution quality.

3.5.2 Pipe Assembly Task

Table 3.4: Statistics For Database Construction (Pipe Assembly)

Metrics	Planning Algorithm						
	RTDP-BEL				E-RTDP-BEL		
	$\varepsilon = 1$	$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$	$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$
Success Rate (%)	71.4	76.5	79.3	81.9	**100**	**100**	**100**
Relative Speedup	1.0	1.4	1.76	2.06	84.44	198.10	**335.50**
Relative Expected Cost	**1.0**	1.06	1.57	1.76	1.08	1.68	1.89

To demonstrate the generality of the framework, the approach was also evaluated in simulation on the task of pipe assembly. Pipe assembly can be modeled as a series of high-precision insertions of pipes into elbows and vice versa (Fig. 3.6). For each insertion in this domain, we reason about the uncertainty in pose along 4 axes (x, y, yaw, and pitch) in contrast to just the positional uncertainty in plug insertion.

As the task is evaluated in simulation, the set of initial uncertainties is artificially created and E-RTDP-Bel is used to create the database of policies. The set contains initial uncertainties ranging up to 2cm in position (along each axis) and 0.25 radians in orientation (along each axis). We used a resolution of 2mm for position and 0.05 radian for orientation for discretizing the distribution. Therefore, $\mathcal{H}$ has hypotheses sets H_i that contain up to 3600 possible poses. During evaluation, random noise is added to the pose of the pipe/elbow. An initial distribution is sampled from the set $\mathcal{H}$ (such that the random noise is contained within the boundary of the discrete distribution), and the corresponding policy is executed by the robot. An important observation is that the random noise is sampled in the continuous space and is not exactly contained in the discrete distribution. The approach was evaluated on 50 problems and we observed that the robot was able to successfully localize the object of interest and complete the task in all of the runs (Table 3.2). Similar to the plug insertion task, Table 3.4 presents the performance of E-RTDP-Bel in constructing the database of policies Π. E-RTDP-Bel continues to provide significant speedups while maintaining good solution quality. The ablation studies and performance comparisons with TBL follow similar trends as in the plug insertion task and are omitted for brevity.

3.6 Conclusion

We formulate high-precision insertion tasks as planning under pose uncertainty and utilize contacts to localize the object of interest and complete the task. We develop a preprocessing-based planning framework for semi-structured insertion domains where the set of possible pose uncertainties are finite and known ahead of time. Due to the computational expense of preprocessing a database of solutions, we propose a general experience-based POMDP solver, E-RTDP-Bel, that effectively utilizes solutions of similar planning problems to speed up planning queries and use it to speed up database construction by over a factor of 100. The framework demonstrates strong performance in terms of robustness, planning times, and solution quality on the tasks of plug insertion (real world) and pipe assembly (simulated).

4 Multi-modal representation for scaling up uncertainties

4.1 Introduction

Chapter 3 addressed the problem of high-precision manipulation under object pose uncertainty in semi-structured industrial settings. By formulating contact-driven localization as a POMDP and exploiting a key source of structure, namely that the set of possible initial pose distributions was finite and known ahead of time, we were able to preprocess a database of solution policies offline. Experience-based RTDP-Bel (E-RTDP-Bel) leveraged similarities across problem instances to make this preprocessing tractable, enabling fast and reliable online execution.

As interest grows in deploying robotic assistants in domestic environments, ranging from early automated vacuum cleaners to more general-purpose household robots [93, 94, 95, 96], robots increasingly encounter settings where these assumptions no longer hold. In domestic environments, relying solely on visual feedback is often inadequate or inefficient. Occlusions, clutter, poor lighting, and constrained viewpoints are common. For example, consider the scenario shown in Fig. 4.1, where a robot must locate an object inside a shelf that is completely out of view. Similar challenges arise when retrieving items from a densely packed refrigerator or during furniture assembly, where one arm stabilizes large components while the other searches for small fasteners. In such situations, humans naturally rely on contact to localize objects and reason about their surroundings. To enable robust object-centric manipulation in domestic settings, robots must be endowed with similar contact-driven reasoning capabilities.

In contrast to the semi-structured industrial setting studied in Chapter 3, the lack of structure in domestic environments manifests in two fundamental ways. First, solution policies must be generated online, as the set of planning problems encountered is neither finite nor known ahead of time. Second, the magnitude of uncertainty that must be resolved is significantly larger.

Domestic robots are required to handle a diverse range of tasks, each with different environmental constraints, object geometries, and task tolerances. Consequently, the sequence of actions required to reduce pose uncertainty depends strongly on the specific problem instance. Enumerating all possible planning problems and preprocessing a database of policies, as done in the semi-structured setting, is therefore infeasible. Instead, the planning framework must synthesize solutions online.

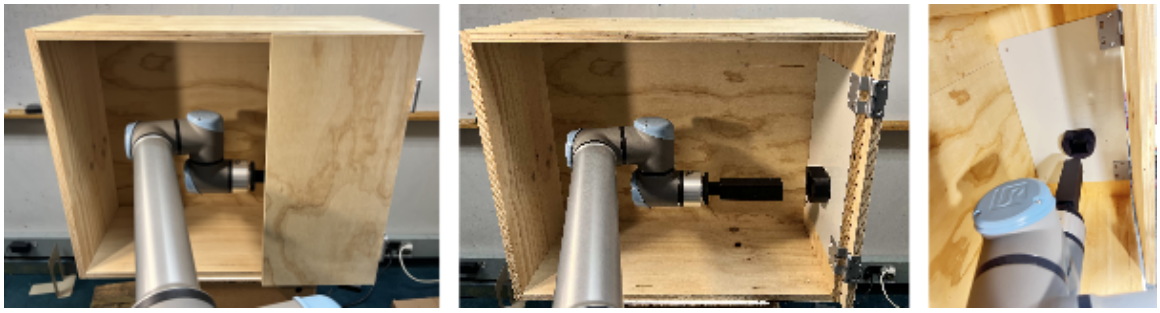

Figure 4.1: Experimental setup for plug insertion using a UR10e manipulator where the port is inside a shelf and out of view of the robot.

However, the POMDPs that arise from contact-based active localization in these settings are highly complex, and solving them exactly or even approximately in real time is computationally intractable without additional structure.

The magnitude of uncertainty further exacerbates this challenge. Returning to the shelf example in Fig. 4.1, since the robot cannot directly observe the object of interest, it must rely on a coarse prior, such as the knowledge that the port is typically located somewhere on the right wall of the shelf. This leads to positional uncertainties on the order of tens of centimeters and angular uncertainty approaching 2π. Similar levels of uncertainty arise when a robot enters a dark room and attempts to locate a light switch near a doorway, where positional uncertainty can approach a meter. These uncertainty scales are orders of magnitude larger than those considered in semi-structured industrial settings.

This increase in uncertainty impacts planning in two critical ways. First, it dramatically increases the branching factor of the underlying POMDP, making search significantly more challenging [13, 97, 98]. Second, belief representation and belief transition computation become prohibitively expensive. Representing belief naively as a particle set would require millions of particles. Evaluating the outcome of a single action would then involve simulating contact interactions for each particle, and solving the POMDP would require rolling out many such action sequences. This makes direct particle-based belief representations impractical in this regime and calls for a more efficient representation.

In this chapter, we address these challenges by developing a framework that incorporates two key ideas:

1. **A hierarchical representation of uncertainty:** We initially represent uncertainty coarsely in a 3D volumetric space that captures regions of the workspace that may be occupied by the object of interest and regions that are guaranteed to be free. Contact and no-contact observations are used to iteratively shrink the possibly occupied volume. Planning is first performed in this volumetric space to reduce uncertainty efficiently. Once uncertainty has been reduced below a predefined threshold, the problem is translated back into a finer particle-based representation by enumerating poses consistent with the remaining volume. Reasoning in the particle space is then tractable, enabling efficient refinement and task completion.

2. **A greedy closed-loop planning and execution framework:** Computing a complete optimal or near-optimal policy for these problems can require hours of computation. Instead, we adopt a greedy closed-loop planning and execution framework with a limited time budget per planning iteration. In each iteration, a heuristic-search-based anytime solver computes a partial policy, which is then executed on the robot. This process is repeated until uncertainty is sufficiently reduced and the task is completed. To maximize performance under tight time constraints, the planner focuses on the most likely outcomes and combines admissible and inadmissible heuristics to balance exploration and exploitation.

We evaluate the proposed framework both in simulation and on a real UR10e manipulator performing plug insertion tasks under large pose uncertainties, including 3D, 4D, and 5D uncertainty. In all cases, the port is positioned outside the robot's field of view, requiring resolution of positional uncertainties on the order of hundreds of centimeters and angular uncertainties close to 2π. Experimental results demonstrate robust performance, achieving a 93% success rate in real-world trials. Performance analyses in Section 6.5 show that the hierarchical belief representation substantially accelerates planning, making online solution synthesis feasible. The proposed planning framework also improves solution quality by up to 30% compared to greedy baselines, with ablation studies highlighting the contribution of each design choice.

4.2 Related work

As outlined in Chapter 3 over the years, various vision-based techniques have been developed for high-precision manipulation tasks such as plug insertions [99, 100, 101]. Visual servoing is a widely used approach for such tasks, utilizing real-time image data from an in-hand camera to compute errors—either directly in image space or by estimating the pose of the insertion slot—and generating control commands to minimize these errors [20, 83, 84, 85].

Many prior works that incorporate tactile feedback focus on developing particle filters and Bayesian estimation methods to compute object poses that best explain the sequence of tactile observations made [38, 46, 47, 48]. However, our work focuses more on active localization using tactile feedback, i.e., reasoning about the sequence of actions to take to quickly localize the object of interest. Previous works tackling this question have utilized a set of particles to represent the object pose uncertainty [14, 51]. Directly formalizing and solving the POMDP by utilizing this representation is infeasible in situations with large uncertainty, prevalent in unstructured environments, as millions of particles may be needed. Other particle filtering variants such as [102] require compute heavy operations like manifold sampling, making them unsuitable for planning. In this work, we address this challenge by proposing a hierarchical representation of uncertainty and defining the POMDP using this representation.

Data-driven end-to-end active perception approaches leverage deep reinforcement learning and neural networks to directly map raw sensory inputs to control actions without explicitly maintaining beliefs [103, 104, 105]. While effective in some tasks, these methods require extensive training data and often generalize poorly to new environments and tasks. Our proposed hierarchical POMDP-

based framework offers a structured way to handle uncertainty while being flexible enough to be integrated with learning-based controllers.

The tasks of peg-in-hole insertion and assembly have been widely studied in robotics, with various strategies explored to handle uncertainty. These approaches include force-based controllers and impedance control strategies [106, 107], model-based methods [108, 109], and, more recently, model-free learning-based techniques [110, 111]. While our framework is demonstrated on high-precision plug insertion, its broader objective is to provide a general real-time POMDP-based planning solution for active localization in uncertain environments. It is even possible to utilize these learned or compliant strategies within our POMDP framework, which methodically reasons about the utility of different action sequences and identifies the one that can most efficiently complete the task.

Our hierarchical representation is independent of the specific planner used to solve the POMDP, allowing for efficient action sequence reasoning. Prior approaches have used greedy methods to approximately solve the POMDP, often employing a myopic framework that interleaves planning and execution [47, 51, 52]. In each iteration of touch-based localization (TBL) [51], actions are sampled, and the robot executes the one that maximizes an information gain metric. This process is repeated until uncertainty is sufficiently reduced. Similarly, frontier search [52] selects the nearest information-gain action for execution. However, these myopic strategies, which focus only on the next best action, often compromise solution quality. Performance comparisons against these baselines presented in Section 5.4 highlight this issue.

The transition models for the proposed representation of uncertainty is closely tied to [36, 38, 39]. However, their objective is to utilize contact observations to model the unknown environment they operate in and efficiently move from one location to another. On the planning front, RTDP-Bel [71] forms the basis of our framework. Further, [112, 113] discuss techniques to focus efforts on the most likely outcomes/beliefs when planning online under time constraints, an idea that we adapt and utilize in our framework.

4.3 Problem Formulation

The problem is formulated similarly to the formulation presented in Chapter 3. Given a robot manipulator, let $\mathcal{Q}$ denote its state space and $\mathcal{A}$ the discrete action space. The observation space $\mathcal{Z}$ consists of:

1. The robot state, obtained from encoder readings, and
2. A binary indicator of contact or collision, detected through force–torque sensor measurements or joint torque signals.

The objective is to compute a robot policy π that utilizes contact observations to reduce uncertainty about the pose of a target object T, ensuring that the remaining uncertainty falls below a task-dependent threshold ϵ. Additionally, the policy should aim to minimize the expected distance traveled while completing the task. The target object T is known to be within a prior volume V in the 3D workspace, ensuring that T is always contained within the initial hypothesis space.

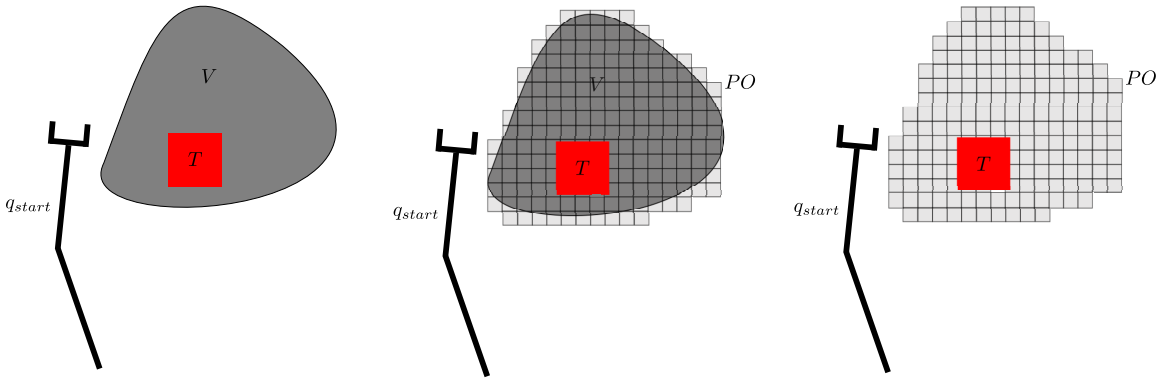

Figure 4.2: Given the target object T (in red) and the space of volume within which the target object is guaranteed to be (in solid grey), PO is a discretized representation of the possibly occupied volume.

The contact observation is a simple binary collision flag that detects if the robot is in contact with an external object, playing a crucial role in refining the belief over T's pose. For example, if a planned motion that avoids the static environment results in a contact observation, it confirms the presence of T along that trajectory. Conversely, if no contact is detected, it eliminates the portions of V that were traversed. Actions that result in contact with the static environment are not informative for reducing uncertainty about T's pose as they are guaranteed to result in a contact observation. Details about the belief transitions and the hierarchical representation of uncertainty are discussed in the section below.

Unlike typical robotics problems, in our domain, the stochasticity in transitions and observations arises from uncertainty in the model of the environment, specifically, the object pose. We assume perfect dynamics and observations, due to the high-quality manipulators and the simple but reliable observation space (contacts) [1]. The environment is assumed to be static with robot actions not altering the state of T. We also assume access to the geometric model of T, which we utilize to compute the expected observations (there are no assumptions on the model of T). It should be emphasized that our approach can be extended to incorporate noisy transition and observation models with minor adjustments. However, in such cases, we anticipate a minor decrease in performance due to the expanded search space.

4.4 Methodology

The planning framework is split into two phases (Fig. 4.3).

Phase 1 (Volumetric representation): We start by representing uncertainty in a 3D volumetric space, which captures the potential volume occupied by the target object. As actions are executed, observations (e.g., collisions) help refine this space. For example, if an action results in no collision, we can infer that the target object does not occupy the swept volume, reducing the possible occupied space. This approach allows for efficient reasoning under large uncertainties and helps

[1] We validated perfect contact observations by sampling 100 actions and confirming that all collision readings aligned with the expected observations.

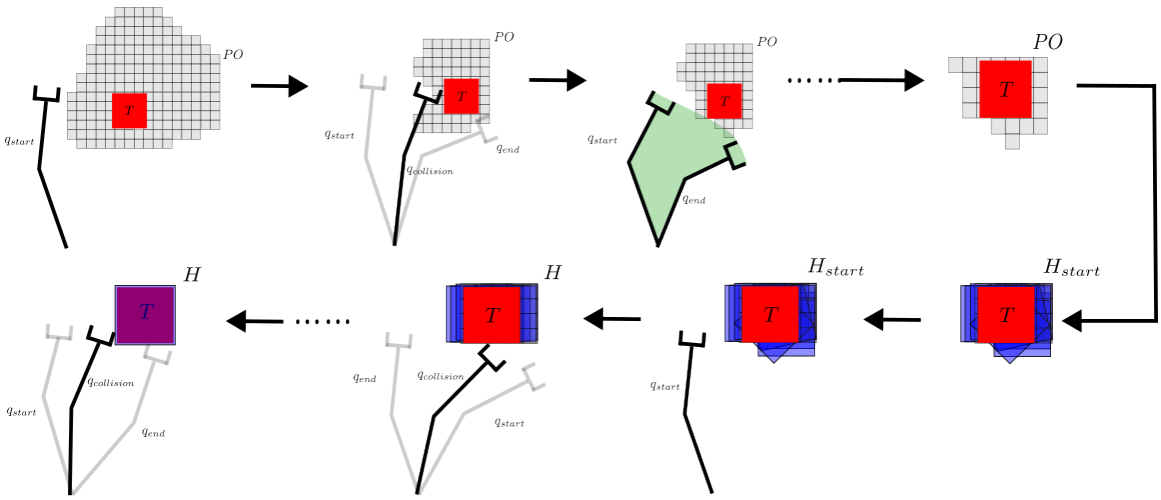

Figure 4.3: The first phase of the framework involves reasoning about pose uncertainty in the volumetric space, using contact observations from actions to reduce the potentially occupied volume. Once uncertainty is reduced to a manageable level, we transition to a finer representation in the particle space by mapping the reduced volumetric uncertainty to a smaller set of feasible poses. We then execute policies that will help reduce uncertainty in this space sufficient enough to complete the task.

identify actions that minimize the potentially occupied volume. A customized heuristic search-based POMDP solver is used to compute a policy that reduces the uncertainty in the volumetric space below a predefined threshold.

Phase 2 (Particle representation): Once the uncertainty is reduced to a manageable level, we transition to a finer representation in the particle space. We map the reduced volumetric uncertainty into a smaller set of feasible target poses (particles). A new POMDP is defined to represent uncertainty transitions in this particle space, and the same heuristic solver is employed to compute a policy that further reduces uncertainty and completes the task.

By hierarchically representing uncertainty, our approach reduces computational complexity, effectively shrinking the search space and allowing a manageable transition from volumetric to particle-based reasoning. The following subsections detail the POMDP formulation in both phases and the customized solver used.

4.4.1 Hierarchical Belief Representation

Volumetric representation

The volumetric representation of uncertainty is structured as a 3D binary occupancy grid, where each voxel has a value of 0 or 1. A voxel with a value of 0 indicates that the object of interest is guaranteed to not occupy any part of it, while a voxel with a value of 1 represents the possibility of occupancy. We denote the set of possibly occupied voxels as PO, where $|PO|$ represents the number of such voxels.

We start by discretizing the initial volume V into a 3D grid, assigning 1 to all cells. As PO is not directly observable, collision observations from executed actions help reduce $|PO|$. The goal of the

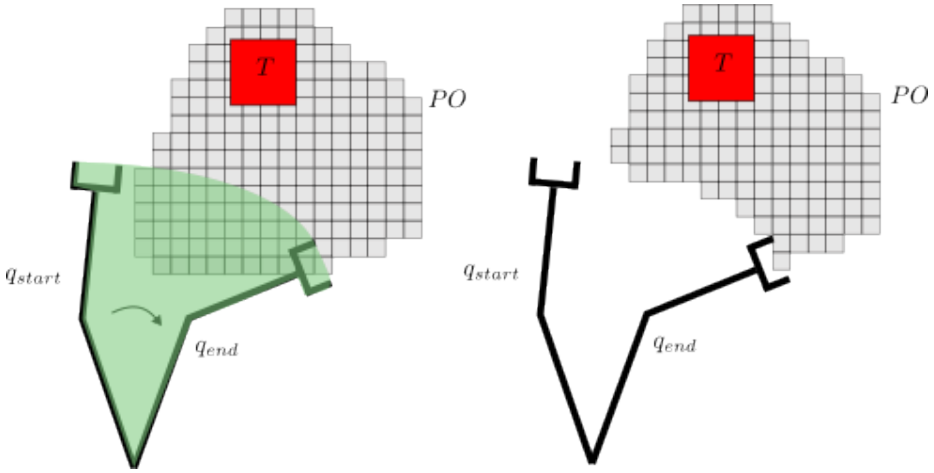

Figure 4.4: If an action results in no contact, the target object is guaranteed to not occupy any part of the swept volume.

planning problem at this phase is to find a robot policy that minimizes the expected travel distance while reducing $|PO|$ below a predetermined threshold δ.

The system's state s comprises the robot's state, the environment state, and the set of previously encountered collisions represented as $s = (q, PO, \mathcal{C})$. Where $\mathcal{C} = \{(q_1, a_1), (q_2, a_2)...\}$ and each tuple $(q_i, a_i) \in \mathcal{C}$ corresponds to the robot having previously observed a collision at configuration q_i while executing a_i. The action space $\mathcal{A}$ consists of end-effector movements. The observation space $\mathcal{Z}$ includes binary collision readings and the robot state $\{(\text{no}) \text{ collision}, q \in \mathcal{Q}\}$. A robot action a starting from q_1 and ending at q_n can be approximated by a discrete set of configurations $\{q_1, q_2...q_n\}$. Such an action can produce $n+1$ possible observations: collisions at any intermediate configuration, $z_i = \{\text{collision}, q_i\}$ or no collision upon reaching q_n, denoted $z_{n+1} = \{\text{no collision}, q_n\}$.

If a collision was observed at q_i when executing a, let $\mathcal{S}(q_i, a)$ represent the set of active voxels on the robot's surface that are potentially in contact with the target object T (Fig. 4.5). This means that at least one voxel in $\mathcal{S}(q_i, a)$ is occupied by T. Additionally, let $\mathcal{D}_{max}$ represent the maximum possible distance between any two points on T and N represent the number of voxels occupied by T.

The successor state s' from executing action a from state $s = \{q, PO, \mathcal{C}\}$, and receiving observation z can be defined as follows:

1. If z_{n+1} is observed (no collision), $s' = \{q_n, PO_{n+1}, \mathcal{C}_{n+1}\}$, where:

$$PO_{n+1} = PO \setminus \mathcal{V}(q_1, q_n); \;\; \mathcal{C}_{n+1} = \mathcal{C} \tag{4.1}$$

   $\mathcal{V}(q_1, q_n)$ represents the set of voxels contained in the volume swept by the robot moving from q_1 to q_n (Fig. 4.4).

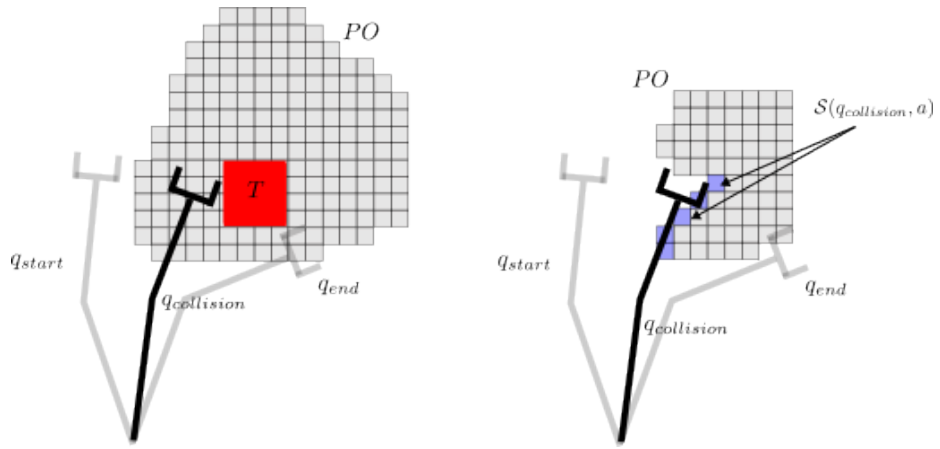

Figure 4.5: If an action a results in contact at configuration $q_{collision}$, the target object is guaranteed to not be farther than $\mathcal{D}_{max}$ from $\mathcal{S}(q_{collision}, a)$.

2. If z_i is observed (collision at q_i), $s' = \{q_i, PO_i, \mathcal{C}_i\}$, where:

$$PO_i = PO \setminus \Big(\mathcal{V}(q_1, q_i) \bigcup \mathcal{U}(q_i, a)\Big)\,;\; \mathcal{C}_i = \mathcal{C} \cup (q_i, a) \tag{4.2}$$

   $\mathcal{U}(q_i, a)$ represents all cells farther than $\mathcal{D}_{max}$ from $\mathcal{S}(q_i, a)$ and are guaranteed to be unoccupied (Fig. 4.5).

The above equations formally establish the notion that if an action observes no collision, then T is guaranteed not to occupy any part of the volume swept by the robot during that motion. Conversely, if a collision is observed at q_i, T is constrained to regions within a bounded distance from q_i, determined by the dimensions of T. Hence, given a state s, action a, and an observation z_i, the corresponding successor state s' can be determined. The probability of observing z_i defines the probability of transitioning to s', denoted as $\mathcal{T}(s', s, a)$. Hence, to define the transition function in the volumetric belief space, the likelihood of each observation z_i, denoted by $\phi(z_i|a, s)$, has to be computed.

Assuming independence amongst the observations, we define the likelihood of collision at configuration q_i as,

$$\phi\Big(\{\text{collision}, q_i\}\Big|PO, a, \mathcal{C}\Big) \;\propto\; \phi\Big(\{\text{collision}, q_i\}\Big|PO, a\Big) \prod_{(q_j, a_j)\in\mathcal{C}} \phi\Big(\{\text{collision}, q_i\}\Big|(q_j, a_j), a\Big) \tag{4.3}$$

The likelihood of not colliding at q_i can be similarly defined. Essentially, the likelihood of observing a collision at q_i while executing a_i is composed of two modules.

First, the likelihood of collision given PO represented by the term, $\phi(\{\text{collision}, q_i\}|PO, a)$. The likelihood of collision depends on the number of occupied voxels in $\mathcal{S}(q_i, a)$. If the active surface of the robot, $\mathcal{S}(q_i, a)$, significantly overlaps with PO, the likelihood of collision is high.

$$\phi(\{\text{collision}, q_i\}|PO, a) = \min(1, \frac{|\mathcal{S}(q_i, a) \cap PO|}{|PO| - N})$$
$$\phi(\{\text{no collision}, q_i\}|PO, a) = 1 - \phi(\{\text{collision}, q_i\}|PO, a) \tag{4.4}$$

Second, the likelihood of collision given the history of observed contacts, $\mathcal{C} = \{(q_1, a_1)...(q_n, a_n)\}$. Given a previously observed collision at (q_1, a_1), the likelihood of observing a collision at (q_i, a) is represented by $\phi(\{\text{collision}, q_i\}|(q_1, a_1), a)$. This likelihood is high if $\mathcal{S}(q_i, a_i)$ intersects with a large portion of $\mathcal{S}(q_1, a_1)$. If $\mathcal{S}(q_1, a_1) \subseteq \mathcal{S}(q_i, a)$, then it guarantees that a collision must be observed at q_i as at least one active surface voxel is guaranteed to be occupied by T. Conversely, the likelihood of a collision decreases as the distance between $\mathcal{S}(q_i, a)$ and $\mathcal{S}(q_1, a_1)$ increases. If this distance exceeds $\mathcal{D}_{max}$, which is the maximum possible separation between two points on T, the likelihood drops to 0.

$$\phi\Big(\{\text{collision}, q_i\} \mid (q_1, a_1), a\Big) = \left(\frac{|\mathcal{S}(q_i, a) \cap \mathcal{S}(q_1, a_1)| + \epsilon}{|\mathcal{S}(q_1, a_1)| + \epsilon}\right)\left(1 - \frac{\max\Big(0,\ \text{dist}\Big(\mathcal{S}(q_i, a), \mathcal{S}(q_1, a_1)\Big)\Big)}{\mathcal{D}_{\max}}\right)$$

$$\phi\Big(\{\text{no collision}, q_i\} \mid (q_1, a_1), a\Big) = 1 - \phi\Big(\{\text{collision}, q_i\} \mid (q_1, a_1), a\Big) \tag{4.5}$$

Finally, these can be combined to compute the likelihood of the different observations z_i. Since the observations are sequential, if z_i is observed, no collision must have occurred at earlier configurations $\{q_1, q_2...q_{i-1}\}$. Thus:

$$\phi\Big(z_i \mid a, s = \{q, PO\}\Big) = \phi\Big(\{\text{collision}, q_i\} \mid PO, a\Big) \prod_{k=1}^{i-1} \phi\Big(\{\text{no collision}, q_k\} \mid PO, a\Big) \tag{4.6}$$

The likelihood of z_{n+1} can also be similarly computed. These likelihoods provide the observation model $P(z|s, a)$, which, together with the state transitions defined by Eqn. 4.1 and Eqn. 4.2, completes the transition model $T(s'|s, a)$. This formalizes the problem in volumetric space, allowing a belief MDP solver to compute a policy that minimizes the expected distance while reducing $|PO|$ below δ.

Particle Representation

After reducing the set of potentially occupied voxels PO, we switch the uncertainty representation to particle space. This involves discretizing the space of possible poses[2] and evaluating each pose to determine if the object at that pose occupies any portion of the workspace outside the potentially occupied voxels set. The object is guaranteed to be completely contained within PO by virtue of construction. Thus, any pose for which the object intersects voxels outside PO is deemed infeasible as it contradicts prior contact observations. Hence, we evaluate all discretized poses and only

[2]The resolution of discretization is dependent on the tolerance of the task.

include the ones that are completely contained within PO into our set of valid hypotheses poses $H_{start} = \{h_1, h_2, ...h_n\}$.

Now that the set of hypothesis poses has been created, we define the POMDP in the particle space similar to [14]. The state of the system in this case contains the robot state and the set of particles $s = \{q, H\}$. An action $a \in \mathcal{A}$ approximated by a sequence of n discrete configurations $\{q_1, q_2...q_n\}$ as previously discussed can result in $n + 1$ observations. Hence, if an observation z_i is made (which corresponds to a collision at q_i), the system transitions to a new state $s' = \{q_i, \hat{H}\}$, where $\hat{H}$ corresponds to all particles (i.e, object poses) in H for which executing action a from q would result in a collision at q_i. The expected contact observation for a given particle h_i is determined by performing collision checks between the discretized approximation of the action and the target object positioned at h_i. Now, given a state s, action a, and observation z_i, it is possible to compute the successor state s'. If the probability of each particle in the set is uniform, then the probability of observing z_i is given by $\frac{|\hat{H}|}{|H|}$ which defines the transition probability $\mathcal{T}(s', s, a)$.

This completes the definition of the POMDP in the particle space (more details can be found in [14]). Now, a belief MDP solver can be used to compute a policy from the start state $s_{start} = \{q_{start}, H_{start}\}$ to a state $s_{goal} = \{q_{goal}, H_{goal}\}$ where s_{goal} satisfies the goal criteria for all the particles in H_{goal}. In the case of the plugin problem, this corresponds to a robot pose s_{goal} such that the charger has been successfully plugged into the port for all poses in H_{goal}.

4.4.2 Planning Framework

Here, we detail the planning framework employed for solving the POMDPs formulated in the previous subsection. We note the overall framework discussed earlier is agnostic to the specific planner used. Our empirical analysis indicate that the planner described below yielded the best results.

Computing complete (bounded sub-)optimal policies for the problems discussed in this manuscript can take extensive compute time (of the order of hours). Since the planner must solve these problems online, we use a closed-loop planning and execution framework with a 1-second time budget for each iteration. In each iteration, we use a modified version of RTDP-Bel (Alg. 5), a heuristic search-based anytime solver to reason over a limited time horizon and compute a partial policy, which is then executed in the real world. If the system reaches a belief state without a precomputed action, the planner is invoked again. This process repeats until the task is completed. To maximize performance, we modify RTDP-Bel in two ways.

Compute actions only for the most probable outcomes

Inspired by existing strategies in literature [112, 113], we optimize the use of limited resources by focusing computational efforts on the most likely scenarios. Specifically, we modify the planner to only compute actions for the most probable beliefs. In each iteration within an episode in RTDP-Bel, the best action from a belief state is computed based on the current value estimates of the successor beliefs from the state. While we consider all possible outcomes when determining the best

Algorithm 5 MODIFIED RTDP-BEL

1: **while** NOT CONVERGED **do**
2: $b = b_{start}$; $depth = 0$
3: **while** $b \notin \mathcal{G}$ and $depth < horizon$ **do**
4: **for** type $\in$ {*admissible*, *inadmissible*} **do**
5: **Evaluate** the value of executing each action $a \in \mathcal{A}$ from belief state b as:

$$Q_{\text{type}}(b, a) = \mathcal{C}(b, a) + \sum_{z \in \mathcal{Z}} P(z \mid b, a)\, V_{\text{type}}(b_a^z)$$

▷ Use $V_{\text{type}}(b_a^z) = \text{heur}_{\text{type}}(b_a^z)$ if not initialized

6: **Update** value of belief state $V_{type}(b) = \min_{a \in \mathcal{A}} Q_{type}(b, a)$
7: **end for**
8: **if** time elapsed $< \epsilon *$ time budget **then** ▷ $\epsilon \in [0, 1]$
9: **Select** action a_{best} that minimizes $Q_{ad}(b, a)$
10: **else**
11: **Select** action a_{best} that minimizes $Q_{inad}(b, a)$
12: **end if**
13: **Pick** most likely b', i.e., $\text{argmax}_{b' \in \mathcal{B}} P(b'|b, a_{best})$
14: **Set** $b := b'$ and $depth := depth + 1$
15: **end while**
16: **end while**

action, the subsequent belief state that we explore is restricted to the most probable successor. In the subsequent iteration the best action for the most probable successor belief is computed (Line 13). If, during the execution of the partial policy, a less likely successor belief is reached, execution is halted, and the planner is invoked again. This approach ensures the planner's limited time is spent reasoning about the most likely situations, enhancing overall effectiveness.

Combining admissible and inadmissible heuristics

The effectiveness of search-based planners relies heavily on the heuristics employed. Heuristics guide the search to explore relevant portions of the belief space, ideally reducing computational costs. An admissible heuristic, which underestimates the optimal cost ($h(b) \leq v^*(b)$), guarantees convergence to an optimal or bounded suboptimal solution. However, constructing effective admissible heuristics is often challenging. An admissible heuristic requires the search to explore all relevant portions of the belief space to guarantee that the optimal solution is not overlooked. This results in large convergence times. On the contrary, inadmissible heuristics tend to be greedy. They encourage the search to quickly converge to a suboptimal solution by exploring a narrow region of the belief space.

In our setting, using only an admissible heuristic resulted in the search extensively exploring and not converging to reasonable partial policies within the limited time budget. Conversely, using an inadmissible heuristic alone led to a very suboptimal solution quickly due to extremely limited exploration. To balance the exploratory and exploitative nature of these heuristics, we use a combination of both admissible and inadmissible heuristics. We maintain both values for a given belief state. When broader exploration is needed, we choose the best action based on the admissible value. For finer optimization, we use the inadmissible heuristic.

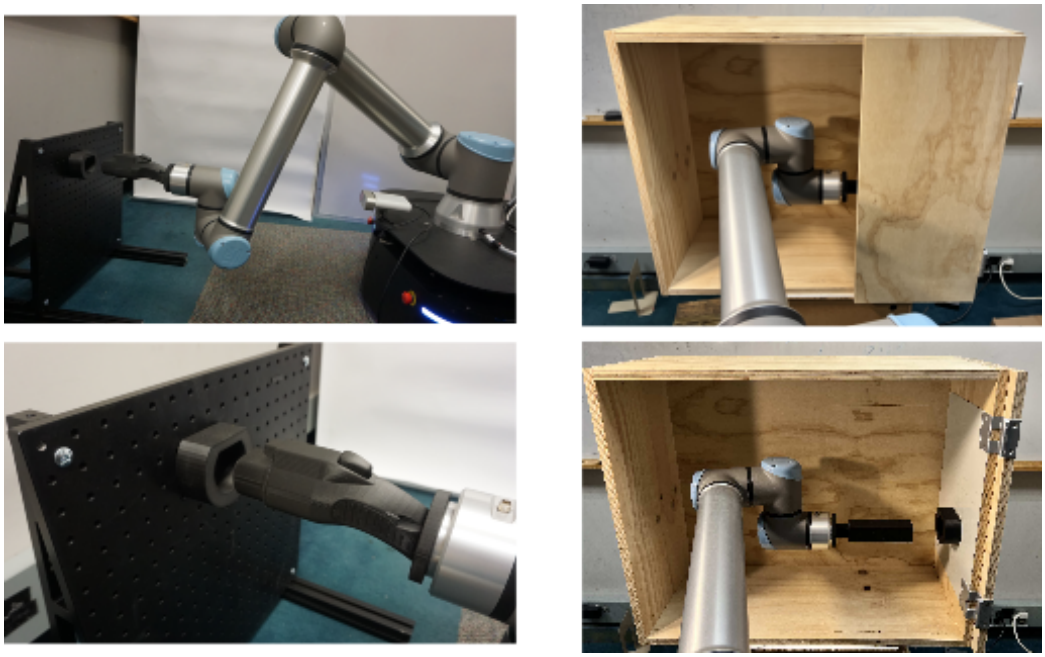

Figure 4.6: Experimental setups. Left) Movable base setup. Right) Shelf setup

We experimented with different schedules for utilizing these heuristics and found that using the admissible heuristic in the initial part of the time budget and leaning on the inadmissible heuristic later proved ideal (Lines 4 - 12). The admissible heuristic promotes exploring different homotopies of the cost manifold, aiming for a global optimum. The inadmissible heuristic, in turn, helps find a local minimum within a homotopy. By exploring various homotopies initially and fine-tuning the solution with the greedy heuristic later, we achieve a balanced and effective search strategy.

4.5 Results

The performance of the overall framework is demonstrated both in real world and in simulation on the task of inserting a plug into a port using a UR10e manipulator under different magnitudes of pose uncertainties. The evaluation includes three different planners: the proposed modified RTDP-Bel and two greedy baselines, TBL [51] and Frontier [52]. TBL is a popular online planning framework that interleaves planning and execution. In each iteration of planning, TBL samples multiple actions and executes the one that maximizes an information gain metric. Based on the observation, the belief of the system state is updated and the process is repeated. On the other hand, Frontier search identifies the nearest action that provides information gain and executes it in the real world. For each iteration of planning, a time budget of 1 second was provided for all of the planners.

Table 4.1: Real-world performance comparisons of the different planners on the plug-in task.

Metric	Ours	TBL	Frontier
Plug Insertion (Shelf)			
Success Rate	28/30	27/30	28/30
Total Time to complete task (s)	33.58	59.96	49.41
Execution time (s)	25.78	54.65	44.59
Planning time (s)	7.80	5.31	4.82
Plug Insertion (Movable Base)			
Success Rate	27/30	28/30	28/30
Total Time to complete task (s)	44.05	83.37	70.26
Execution time (s)	33.60	76.61	63.84
Planning time (s)	10.45	6.76	6.42

4.5.1 Real World Robustness

The planner's performance was evaluated in the real world under two different setups. In the first setup, the robot is tasked with localizing a plug on the wall of a shelf, with its exact pose unknown due to obstruction (Fig. 4.6). This involves in-plane localization ($SE(2)$), where the robot must resolve 3-dimensional pose uncertainty $(y, z, roll)$, with positional uncertainty of 50 cm and angular uncertainty of 2π. In the second, the port was mounted on a movable base, requiring the robot to resolve 4-dimensional pose uncertainty (x, y, z, yaw), with positional uncertainties of 40 cm and yaw uncertainties of 50 degrees. The framework was evaluated on 30 real-world runs for each setup, with the port placed in random poses within its workspace. The robot's start state was defined such that the end effector was positioned at a distance of 10 cm along the x-axis from the initial hypothesis volume V [3].

The results presented in Table 4.1 show that the planners succeed in 28 of the 30 runs on average highlighting the robustness of the framework for a high-precision task like plug insertion that has very low tolerance to pose estimation errors (errors larger than 3mm in position can cause task failure). We observed that our proposed planning framework was able to complete the task significantly faster in comparison to the baseline planning approaches. Table 4.1 also details the division of time between planning and execution. While our planner requires slightly more planning time, it identifies significantly better policies, leading to substantial overall savings in the total time required to complete the task. The reduced planning times of the baselines are due to their use of only a portion of the time budget. Frontier search exits as soon as it identifies the nearest information gain frontier, incurring the least planning time. TBL evaluates a fixed number of actions in each planning cycle, and as uncertainty reduces, the time required to evaluate actions decreases, which lowers the average planning time. We also evaluated a variant of TBL that evaluates actions until the time

[3]Since reasoning about contacts and lack of contacts only happens within the hypothesis volume, the start state did not have any major impact on performance and was set at a fixed distance from V.

Table 4.2: Simulation performance comparisons of the different planners on the plug-in task (all times are presented in seconds).

Planner	Total time	Execution time	Planning time	Num iters	Plan Time per iter
Overall Performance					
Ours	52.56	40.27	12.29	16.70	0.73
TBL	94.22	85.37	8.85	27.71	0.32
Frontier	79.98	72.88	7.11	38.60	0.18
Volumetric Space Performance					
Ours	40.74	31.81	8.93	12.53	0.71
TBL	68.03	61.91	6.12	21.06	0.29
Frontier	62.15	56.66	5.49	32.14	0.17
Particle Space Performance					
Ours	11.81	8.46	3.35	4.17	0.80
TBL	26.18	23.46	2.72	6.65	0.41
Frontier	17.83	16.22	1.61	6.46	0.25

budget expires. However, this variant resulted in similar execution times but higher planning times.

The few observed failures occurred when our method eliminated all possible particles. This happened because our initial hypothesis set, H_{start}, was generated by sampling at a discretized resolution, and the true pose was not sufficiently close to any sampled particle. Given the task's low tolerance, we used a resolution of 3 mm in position and 5 degrees in angle. Increasing the sampling resolution to 1 mm and 2 degrees resolved the issue, raising the success rate to 100%. However, this also increased execution and planning times by roughly 30% and 35%, respectively, across all planning algorithms. Hence, denser sampling can be employed when the current system fails, albeit at the expense of longer task completion times.

4.5.2 Simulation Comparisons

We observe that the time required to evaluate 10 actions increases (roughly) linearly with the number of particles, reaching 7 seconds for 5000 particles and over 60 seconds for 40000. This demonstrates the necessity of the hierarchical representation of uncertainty, as evaluating even 10 actions becomes impractically slow when the number of particles exceeds 5000. Given the large number of particles typically involved (in the millions) and the need to evaluate thousands of actions per planning cycle, direct particle space reasoning is infeasible. It also must be reiterated that the proposed hierarchical representation is independent of the planner employed and makes planning under large magnitudes of pose uncertainty feasible.

Table 4.3: Evaluating the impact of combining admissible and inadmissible heuristics (all times are presented in seconds).

Planner	Total Time	Execution Time	Planning Time	Num Iters	Plan Time per Iter
Ours	56.69	42.38	14.31	18.2	0.79
RTDP-Inad	73.13	60.53	12.59	20.93	0.60

Table 4.2 presents the framework's performance in simulation for the plug insertion task under 5D pose uncertainty $(x, y, z, roll, yaw)$. The positional uncertainty is approximately 30 cm, while the angular uncertainty is around 30 degrees. The total time represents the sum of execution and planning time required to complete the task. The baseline planners exhibit 1-step greediness, leading to higher execution times, which our proposed planner reduces by more than 50%. Frontier search has the shortest per-iteration planning time as expected. Although our planner requires slightly more planning time, it significantly reduces execution time, completing the task much faster than the baselines. We also analyze the planners' performance in both the particle space and the volumetric space. Our framework consistently outperforms the baselines in execution time, achieving reductions of up to 2.8 times in the particle space and 1.9 times in the volumetric space. Since a substantial portion of the problem is addressed in the volumetric space, our framework achieves an overall execution time improvement of up to 2.12 times. Additionally, we report the number of iterations each planner takes and the average planning time per iteration.

Table 4.3 illustrates the impact of combining admissible and inadmissible heuristics. Using only an inadmissible heuristic causes the search to quickly converge to suboptimal solutions. This is reflected in the higher execution times and shorter planning times. In contrast, combining it with an admissible heuristic improves solution quality by up to 40% by enabling exploration of a larger portion of the space. Solely using an admissible heuristic results in excessive exploration of the belief space, failing to identify effective policies within the time budget and completing the task. Due to its very low success rate, its results are not included in the table.

4.6 Conclusion

In this work, we presented an online planning framework for unstructured settings, enabling robots to use binary contact signals to reduce large pose uncertainties and complete manipulation tasks. To manage the computational complexity, we employ a hierarchical approach. Initially, we reason in a coarse 3D volumetric space; once uncertainty is sufficiently reduced, we transition to particle space for finer adjustments. This method significantly reduces planning complexity, enabling real-time problem solving under large uncertainties. Our closed-loop framework uses a heuristic search-based anytime solver to compute partial policies within a limited time budget. Our approach achieved a 93% real-world success rate and improved solution quality by over 50% compared to greedy baselines on a high-precision task.

5 Lazy heuristic search for planning with expensive belief transitions

5.1 Introduction

Chapters 3 and 4 demonstrated that contact-driven planning under pose uncertainty can be effectively formulated and solved as POMDPs using heuristic search in belief space. In semi-structured settings, this formulation enabled offline preprocessing of solution policies by exploiting a finite and known set of uncertainty distributions. In contrast, in domestic settings with significantly less structure and larger uncertainty, the same formulation supported online planning through hierarchical belief representations and greedy closed-loop planning and execution.

Both frameworks rely fundamentally on search-based POMDP solvers such as RTDP-Bel [71] and LAO* [72], which use heuristics to focus computational effort on promising regions of the belief space and converge to optimal or bounded suboptimal solutions. While powerful and effective, a critical assumption underlying the computational efficiency of these algorithms is that belief transitions are readily available, meaning that they can be computed cheaply whenever required during search.

In a POMDP, the true system state is not directly observable, and uncertainty is represented as a probability distribution over states, referred to as a belief. Planning is therefore performed in the belief space by formulating the problem as a Markov Decision Process over beliefs. Solving POMDPs is known to be challenging due to the exponential growth of the belief space with both the number of state variables and the planning horizon, commonly referred to as the curse of dimensionality and the curse of history [10]. While these challenges have been studied extensively, a comparatively underexplored issue with significant computational consequences in robotics is the cost of computing belief transitions.

Computing a belief transition requires determining the successor beliefs and their probabilities given a belief and an action, which involves Bayesian updates combining the transition and observation models of the underlying POMDP. In many robotics domains, including contact-rich manipulation, these models are expensive to evaluate. In the context of manipulation under object pose uncertainty using contacts, computing belief transitions requires performing mesh-to-mesh collision checks to model expected contact interactions. Similarly, outdoor navigation may require

high-fidelity physics simulation to predict action outcomes, while indoor navigation often relies on expensive ray casting operations to simulate onboard sensors. These operations are costly even for a single system state. Since a belief represents a distribution over many states, computing a single belief transition requires evaluating these expensive operations for every state in the belief support. When heuristic search explores thousands of belief states and evaluates multiple actions at each belief, the cumulative cost of belief transition computation quickly becomes prohibitive, severely limiting the practical applicability of POMDP-based planning in such domains.

In this chapter, we address this bottleneck by introducing lazy heuristic search methods for POMDPs that defer expensive belief transition computations until they are necessary. Specifically, we propose two algorithms, Lazy RTDP-Bel and Lazy LAO*, which leverage Q-value estimation to identify promising actions from a belief state and selectively compute belief transitions only for those actions. Belief transitions for less promising actions are postponed until the search determines that they may improve the current solution. By avoiding unnecessary belief transition evaluations, the proposed methods significantly reduce computational overhead. We show that the Q-value estimates used by our lazy planners are equivalent to heuristic functions, are readily available for our contact-based planning problem as well as other common robotics domains, and that when these estimates are conservative, the lazy planners retain the same optimality guarantees as their non-lazy counterparts.

We demonstrate that the proposed lazy planners speed up planning for contact-based object pose localization by up to 2× compared to standard heuristic search solvers. To highlight the generality of the approach, we also evaluate the algorithms on additional robotics problems, including indoor navigation with a 1D LiDAR sensor and outdoor rough-terrain navigation. In each of these domains, belief transition computation is expensive due to the need for collision checking, ray casting, or physics-based simulation. Results presented in Section 5.4 show that lazy heuristic search significantly improves planning efficiency while maintaining solution quality across all evaluated tasks.

5.2 Related Work

Beyond heuristic search, two widely studied approaches for scalably solving POMDPs are point-based methods and Monte Carlo methods. Point-based POMDP solvers are a class of approximate algorithms designed to efficiently solve large POMDPs by selectively maintaining and updating a representative set of belief points. Solvers such as PBVI [74], HSVI [75], and SARSOP [114], leverage value iteration on a subset of beliefs to compute near-optimal policies while significantly reducing computational complexity. However, Point-based backups—a core operation of these solvers—assume the transition and observation models are readily accessible and require querying them at every state in the state space. While feasible for simpler problems, where one can precompute a tabulated transition and observation matrix, our domains involve large state spaces with expensive models, making such enumerations impractical. Extensive discussions on the many point-based solvers can be found in [115]. On the other hand, Monte Carlo methods, particularly suited for online planning, approximate the value of belief states by simulating action sequences and prop-

agating rewards. Works like [116, 117, 118] sample state trajectories and update action values based on rollouts. DESPOT [119], a popular approach in this realm, improves efficiency by maintaining a fixed number of sampled scenarios, generating a sparsely sampled belief tree that performs well in problems with large observation spaces.

In deterministic search, the concept of laziness has been widely explored to defer checking the validity of edges, which is often the computational bottleneck, especially in robotics applications where collision checking is expensive. Lazy Weighted A* (LWA*) [120] postpones edge evaluations until expansion, meaning only edges leading to expanded nodes are evaluated. On the other hand, LazySP [121] initially assumes all edges are valid and finds a path to the goal; if an edge is found to be invalid, it is removed from the graph, and the search is rerun, minimizing the number of evaluations. Variants presented in [122, 123, 124] further leverage laziness to accelerate planning. While prior work has focused on deferring edge evaluations in deterministic settings, we extend this concept to stochastic domains by postponing expensive belief transition computations.

5.3 Methodology

Many heuristic search methods for POMDPs are on-policy, exploring only belief states in the current best solution policy [71, 72, 125]. They maintain Q-values for actions available from each encountered belief state. At a given belief b, the best action a_{best}, minimizing the Q-values is selected, and the search explores only its successor beliefs. The Q-value of a_{best} is then updated based on the outcome of this exploration. If this update changes the minimizing action at b, a_{best} is updated, and the search proceeds with the new best action and its corresponding successor beliefs.

The key observation here is that, since only the successor beliefs of the current best action a_{best} are relevant and explored, belief transitions for all other actions remain unused. Consequently, computing these unused transitions introduces unnecessary computational overhead. Hence, we propose to only evaluate the best action a_{best} from any explored belief state. By generating belief transitions only for a_{best}, we defer computing the belief transitions for all other actions until the search deems these actions to be promising, i.e., has potential to be part of the solution policy. This ensures that belief transition evaluations are performed only when necessary, significantly reducing computational complexity. Actions that are never considered promising at any point during the search remain unevaluated. This key principle of postponing action evaluations until they are deemed useful forms the essence of lazy search.

However, existing methods typically initialize Q-values using a one-step lookahead (Alg. 1 Line 5; Alg. 2 Line 5), which involves evaluating each action by computing belief transitions and using successor belief values (initialized with a heuristic) to determine the actions' Q-values. This process incurs expensive belief transition computations for all actions, which the lazy search aims to defer. Hence, we propose replacing the one-step lookahead with a fast-to-query estimator that does not require belief transition computations. We denote this estimator as $\hat{Q}_{init}$, as it is used solely for initializing Q-values. Under this approach, only the belief transitions for a_{best} are computed.

The Q-estimators our lazy planners leverage are similar to heuristic functions and can be easily constructed for common problems. Just as search algorithms like RTDP-Bel and LAO* converge

Algorithm 6 LAZY RTDP-BEL

```
1:  procedure LAZY RTDP-BEL
2:      while NOT CONVERGED do
3:          b = b_0
4:          Sample state s with probability b(s)
5:          while b is not a goal belief do
6:              if Q-values not initialized for b then
7:                  Set Q(b, a) = Q̂_init(b, a)  ∀a ∈ A
8:              end if
9:              do                                   ▷ Do while loop (e.g., to Line 13)
10:                 Select a_best that minimizes Q(b, a)
11:                 Evaluate a_best from b if unevaluated
12:                 UPDATE Q-VALUES(b)
13:             while a_best ≠ argmin_{a∈A} Q(b, a)
14:             Update value V(b) = Q(b, a_best)
15:             Sample s' with probability T(s, a_best, s')
16:             Sample z with probability O(s', a_best, z)
17:             Compute b^z_{a_best}; set b := b^z_{a_best} and s := s'
18:         end while
19:     end while
20: end procedure

21: procedure UPDATE Q-VALUES(b)
22:     for a ∈ A do
23:         if a evaluated from b then
24:             Q(b, a) = C(b, a) + Σ_{z∈Z} P(z|b, a)V(b^z_a)
25:                                                  ▷ V(b^z_a) = heur(b^z_a) if uninitialized
26:         end if
27:     end for
28: end procedure
```

optimally with an admissible heuristic, $\text{heur}(b) \leq V^*(b)$, their lazy counterparts guarantee optimality if $\hat{Q}_{init}$ is admissible, i.e., $\hat{Q}_{init}(b, a) \leq Q^*(b, a)$.

We first introduce the lazy adaptations of RTDP-Bel and LAO*, followed by a brief discussion of simple techniques for constructing effective estimators.

5.3.1 Lazy RTDP-Bel

The lazy adaptation of RTDP-Bel is outlined in Algorithm 6 and is fairly straightforward with key differences being from Lines 7-13. When a belief state is encountered for the first time, the Q-values of all available actions are initialized using the fast-to-compute estimator, $\hat{Q}_{init}$ (Line 7). The algorithm then identifies the best action, i.e., the action that minimizes the Q-values. Unlike standard RTDP-Bel, only this best action is evaluated, meaning that the belief transitions corresponding to the selected action are computed. The Q-value of the evaluated action is then updated accordingly through routine UPDATE Q-VALUES. If this update results in a change in the best action, a_{best} is updated, and the process is repeated until the best action remains unchanged following an update

Algorithm 7 Lazy LAO*

```
1:  procedure LAZY LAO*
2:      G* ← {b_0}
3:      while G* has non-terminal tip states do
4:          Identify a non-terminal tip state b in G*
5:          if Q-values not initialized for b then
6:              Set Q(b,a) = Q̂_init(b,a)  ∀a ∈ A
7:          end if
8:          do                                          ▷ Do while loop (e.g., to Line 12)
9:              Select action a_best that minimizes Q(b,a)
10:             Evaluate a_best from b if unevaluated
11:             UPDATE Q-VALUES(b)
12:         while a_best ≠ argmin_{a∈A} Q(b,a)
13:         Update value V(b) = Q(b, a_best)
14:         Create set Z with b and ancestors of b in G*
15:         IMPROVEVALUES(Z)
16:         Update G* to current best solution
17:     end while
18:     IMPROVEVALUES(G*)
19:     if G* ≠ current best solution then
20:         Update G* to current best solution
21:         Goto Line 3
22:     end if
23:     return G*
24: end procedure

25: procedure IMPROVEVALUES(B)
26:     while Error bound greater than ε do
27:         for b ∈ B do
28:             UPDATE Q-VALUES(b)
29:             Select action a_best that minimizes Q(b,a)
30:             if a_best not evaluated from b then
31:                 return
32:             end if
33:             Update value V(b) = Q(b, a_best)
34:         end for
35:     end while
36: end procedure
```

(Lines 9–13). This approach ensures that the algorithm evaluates only the actions that minimize the Q-value for any belief state encountered. By deferring the evaluation of actions until they are deemed promising and potentially part of the (bounded sub-)optimal solution, the method efficiently reduces unnecessary computations.

5.3.2 Lazy LAO*

The lazy adaptation of LAO* outlined in Alg 7 follows the same fundamental structure as vanilla LAO*. Like its standard counterpart, the algorithm maintains the current best partial solution graph, G^*, and iteratively identifies non-terminal tip states for expansion. However, we slightly modify the definition of a non-terminal tip state. In vanilla LAO*, a non-terminal tip state is a non-goal belief whose actions have not been evaluated. In our adaptation, a non-terminal tip state is defined as either (i) a non-goal belief with no evaluated actions or (ii) a non-goal belief whose best action—the action with the lowest Q-value—has not yet been evaluated.

Using this modified definition, we identify a non-terminal tip state in G^* (Line 4). If the belief state has not been encountered before, its Q-values are initialized using the quick-to-compute estimator $\hat{Q}_{init}$ (Line 6). As in Lazy RTDP-Bel, the algorithm identifies the action minimizing the Q-values and evaluates only this action. The evaluated action's value is then updated accurately. If this update changes the best action, the newly identified best action is evaluated, and the process repeats (Lines 8-12). This ensures only promising actions from a state are evaluated. After this step, the algorithm proceeds like LAO*, using the Q-value of the best action to update b. The set Z, consisting of b and its ancestors within G^*, is then constructed, and value improvements are performed. This continues until no non-terminal tip states remain in G^*. At this point, the values of all states in G^* are further refined until either G^* is modified or the values converge (Lines 13-23).

A key difference from vanilla LAO* lies in how values are improved in the ImproveValues routine. In standard LAO*, this routine is simple value iteration, assuming all actions from a belief state are either evaluated or can be cheaply evaluated. In contrast, our adaptation accounts for lazy evaluations. If, during backup, the best action of a state changes to an unevaluated one, backups terminate (Line 30). If the best action for any state in either set Z or G^* changes, it implies that the current best partial solution has been modified, necessitating an update to G^* (Lines 16, 20), followed by which the algorithm returns to Line 3 (Lines 17, 21). Notably, actions are evaluated only in Line 10; at no other point are they explicitly evaluated. These modifications ensure that action evaluations are deferred as much as possible, providing sizeable computational savings.

Theorem 5.3.1. *If the estimator $\hat{Q}_{init}$ is an underestimate, i.e., $\hat{Q}_{init}(b,a) \leq Q^*(b,a)\ \forall\ b \in \mathcal{B}, a \in \mathcal{A}$, where $Q^*(b,a)$ corresponds to the optimal Q-value, then both Lazy RTDP-Bel and Lazy LAO* will converge to the optimal policy.*

Proof. The idea is similar to the optimality proof for vanilla RTDP-Bel and LAO*. Since Q_{init} is an underestimate, Bellman backups ensure $Q(b,a) \leq Q^*(b,a)$ for all belief-action pairs encountered throughout the search. Each backup progressively increases the value of the current best action (the action that minimizes Q-values) at belief states along the current policy, bringing it closer to the optimal value. If a suboptimal action initially minimizes the Q-value at a belief state, updates eventually raise its value above that of the optimal action, ensuring the correct action is selected and the search converges to the optimal policy. □

Full horizon laziness

While the bottleneck in computing belief transitions is typically expensive transition or observation models, in some planning problems (as shown in Section 5.4), the bottleneck is verifying an action's validity from a belief state. Here, successor beliefs can be efficiently computed, but determining action feasibility is computationally expensive. For example, in navigation tasks, an action may be deemed invalid from a belief b if executing it from any state s with $b(s) > 0$ could cause a collision. Verifying feasibility requires expensive collision checks, while computing successor beliefs assuming the action is valid is relatively cheap (as motion-primitives are known). In such cases, we can adopt what we call full-horizon lazy planning. Here, the policy is initially computed (using your favorite solver) assuming all actions are valid. Only actions in the computed policy undergo the expensive validation process. Invalid actions are marked infeasible, and the planner is queried again. This repeats until the policy consists entirely of valid actions. Similar to LazySP in deterministic settings, this approach defers the bottleneck operation until a complete policy is computed, minimizing the number of expensive queries. However, it is only applicable when successor beliefs can be computed efficiently and the primary computational cost lies in verifying action feasibility.

5.3.3 Q-value estimators for lazy planning

Since the estimator $\hat{Q}_{init}$ is only used to initialize the Q-values, it can be interpreted as an approximation of the initial Q-values, $Q_{init}(b, a)$, defined as

$$Q_{init}(b, a) = \mathcal{C}(b, a) + \sum_{z \in Z} P(z|b, a) \cdot \text{heur}(b_a^z). \tag{5.1}$$

A strong correlation between $\hat{Q}_{init}$ and Q_{init} enhances the effectiveness of lazy planners. To guarantee optimality, $\hat{Q}_{init}$ should underestimate Q_{init} (assuming the heuristic is admissible). However, a close approximation suffices for strong performance. Prior work in deterministic planning has explored using learned models to predict heuristic values [126, 127]. We see potential in extending these ideas to stochastic settings. Domain-specific knowledge can also be leveraged to construct useful estimators.

Our focus is not on designing sophisticated estimators but on demonstrating the utility of lazy planners when paired with an estimator. In addition to the approaches mentioned, we present a simple domain-independent subsampling method that proved effective in our empirical analysis.

The key computational bottleneck in computing $Q_{init}(b, a)$ (Eq. 5.1) is determining successor beliefs b_a^z. This requires evaluating action a on belief b, which is expensive as b is a distribution over the state space $\mathcal{S}$. The action must be evaluated for every state s where $b(s) \neq 0$.[1] Given the number of belief states typically encountered during search, the computational cost is significant.

To address this, we estimate $Q_{init}(b, a)$ using subsampling. We construct a discrete approximation $\hat{b}$ of b by sampling states $s \sim b$ until $\text{supp}(\hat{b})$ reaches a predefined threshold Δ. The frequency of occurrence of each sampled state determines its probability in $\hat{b}$. Since $Q_{init}(\hat{b}, a)$ can be computed

[1] In many robotics problems, the state space is large. While the support of a typical belief state during search is substantial, it still represents only a fraction of the total state space.

efficiently due to the reduced support and approximates $Q_{init}(b, a)$ well, we define the subsampled estimator as $\hat{Q}_{init}(b, a) = Q_{init}(\hat{b}, a)$.

The subsampling estimator in its vanilla form is effective when the heuristic satisfies $\text{heur}(b) = \mathbb{E}_{s\sim b}\text{heur}(s)$, a property common in robotics planning, especially in goal-directed tasks (e.g., an autonomous vehicle reaching a goal). However, if the heuristic does not satisfy this property——such as when it measures belief entropy (e.g., in active localization, where the cost of reducing uncertainty correlates well with the current uncertainty level [128])—correction terms can be introduced to mitigate bias. Although effective, the subsampled estimates are not guaranteed to be conservative. A detailed discussion of the subsampling estimators for our problem and the other evaluated domains are presented in Section 8.

Additionally, in problem domains where belief transition costs stem primarily from the observation model, while querying the transition model is inexpensive, we propose an estimator inspired by Q^{MDP} heuristics [129]. Commonly used as admissible estimates in POMDP planning, Q^{MDP} heuristics assume the state becomes fully observable after a single action. We define this estimator as:

$$\hat{Q}^{MDP}_{init}(b, a) = \sum_{s\in\mathcal{S}} b(s)\left(c(s, a) + \sum_{s'} \mathcal{T}(s, a, s')\text{heur}(s')\right)$$

The estimator $\hat{Q}^{MDP}_{init}$ is conservative if $\text{heur}(s')$ underestimates the cost of reaching the goal set G from s' in the underlying MDP defined by the transition model $\mathcal{T}$.

5.3.4 Discussion

The idea of lazily evaluating belief transitions by deferring their computation until necessary extends well beyond the two specific instantiations discussed above. It naturally applies to variants such as ILAO* [72], IBLAO* [130], and AO* [131]. POMHDP [78], which augments RTDP-Bel with multiple heuristics, can also incorporate laziness by using separate Q-estimators for each heuristic. Beyond these, the concept extends to other heuristic search methods like AEMS2 [125], BI-POMDP [132], and [112]. With a few modifications, we also see potential in extending it to off-policy algorithms like AEMS [125], which expand belief nodes outside the current best policy. Overall, lazy heuristic search for POMDP planning forms a broad class of algorithms. Finally, the core idea of deferring transition computations via Q-value estimation can be applied to MDPs as well.

5.4 Results

We evaluate the performance of the proposed lazy planners on the problem of manipulation for object pose estimation using contacts. To demonstrate the generality of the approach beyond contact-rich manipulation, we additionally evaluate the algorithms on two other robotics domains: indoor navigation with a 1D LiDAR sensor and outdoor rough-terrain navigation. Across all do-

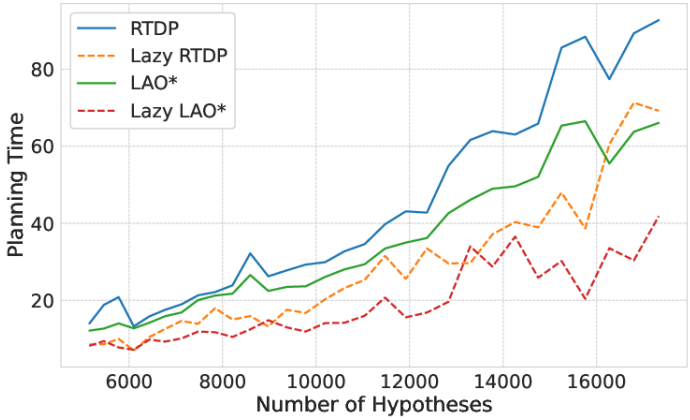

Figure 5.1: Planning times as a function of object pose uncertainty on the manipulation for pose estimation problem.

mains, beliefs are represented as discrete particle sets, and observations are assumed to be perfect.[2] For all experiments, the fast-to-compute Q-value initializer is implemented using a subsampling estimator of the form

$$\hat{Q}_{init}(b, a) = Q_{init}(\hat{b}, a),$$

where $\hat{b}$ is a subsampled belief constructed by randomly selecting a fixed fraction of the particles in b. Unless stated otherwise, $\hat{b}$ contains 15% of the particles in the full belief, corresponding to $\Delta = 0.15$.

Reported planning time and solution cost statistics are averaged over problem instances for which at least one algorithm successfully solves the problem. To ensure fair and representative comparisons, we exclude statistics for any algorithm that succeeds on fewer than 20% of the evaluated problem instances within the allotted computational budget.

5.4.1 Manipulation for pose estimation using contacts

Planner	Success Rate	Planning Time	Cost
RTDP-Bel	0.87	307.49	0.69
Lazy RTDP-Bel	1.0	183.18	0.71
LAO*	0.93	253.67	0.69
Lazy LAO*	1.0	136.73	0.70

Table 5.1: Performance comparison on the manipulation for pose estimation problem.

[2]This assumption is made for clarity and consistency with prior chapters and is not a limitation of the proposed framework. The lazy planners directly extend to other belief representations and to noisy transition and observation models.

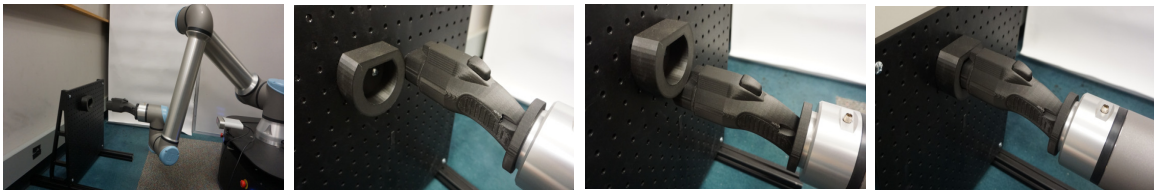

Figure 5.2: The robot utilizing contact observations from different actions to reduce uncertainty about the pose of the object of interest, i.e., the charger port, before completing the insertion task.

The problem formulation considered in this chapter is identical to that introduced in Chapter 3. The objective is to localize a target object T (a charger plug) and subsequently complete a high-precision insertion task. The pose of the plug is not directly observable. Instead, the robot manipulator infers the pose through binary contact observations obtained during execution (Fig. 6.7). For example, if no contact is observed while the robot sweeps a region of space, the object is guaranteed to be absent from that region. Conversely, observing contact confirms the presence of the object within the swept volume.

Uncertainty over the object pose is represented using a discrete particle set $H = \{h_1, h_2, \dots\}$, where each h_i corresponds to a feasible hypothesis pose of the target object. The action space consists of unit robot motions as well as compliant control actions that allow the robot to track object surfaces upon contact. The observation space includes the robot configuration q, obtained from joint encoders, and a binary collision signal derived from force-torque sensor readings. The belief state is therefore represented as $b = \{q, H\}$.

The planning objective is to compute a policy that fully localizes the target object, that is, reduces the hypothesis set to a single element ($|H| = 1$), while minimizing the robot's expected travel distance. For a given object pose h_i and action a, computing the expected observation requires forward simulating the action with the object model placed at h_i. This involves performing expensive mesh-to-mesh collision checks to determine the sequence of contact events and robot configurations encountered during execution. Consequently, computing successor beliefs from a belief state $b = \{q, H\}$ requires simulating the same action for every hypothesis pose $h \in H$, making belief transition computation particularly expensive.

A detailed exposition of the problem formulation, including the belief update equations and transition model, can be found in Chapter 3. Since this is an active information-gathering problem, we define the belief heuristic as a scaled function of the cardinality of the hypothesis set H, which serves as a proxy for belief entropy. To initialize Q-values efficiently, we employ the subsampling estimator introduced earlier, with a minor modification to ensure unbiased estimation. Details of this modification are provided in Section 8.

Results averaged over 100 runs are presented in Table 5.1. In each run, the planner resolves 3D positional uncertainties in the pose of T, with uncertainty along each axis randomly chosen between 4mm and 80mm. This uncertainty is discretized at a 2mm resolution to form the hypothesis set H, meaning a 30mm uncertainty per axis results in a start belief with 15^3 hypotheses. Each planner was given a 500-second timeout. The results show lazy planners outperform their vanilla counterparts by solving problems more efficiently through fewer belief transition evaluations. Lazy planners incurred

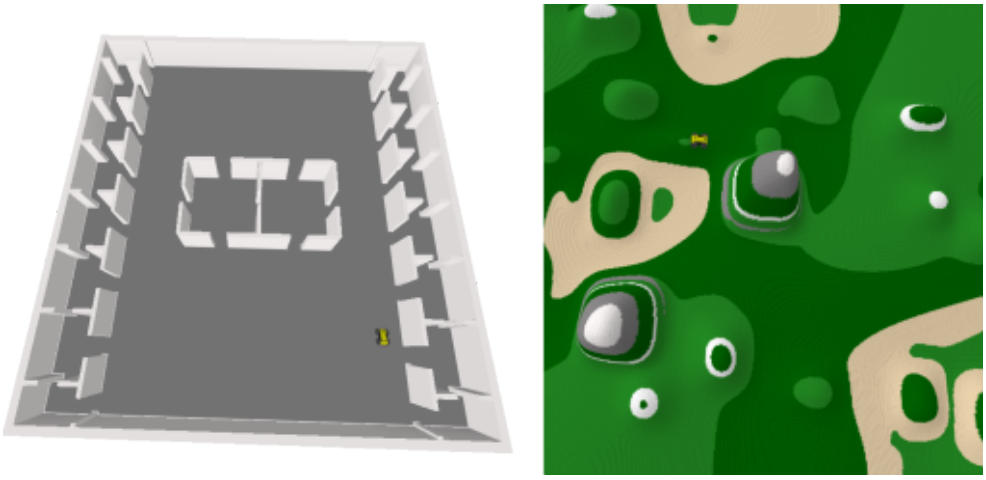

Figure 5.3: Visualization of the indoor (left) and outdoor navigation environments (right).

roughly 50-60% of the planning times of their vanilla counterparts, enabling them to tackle harder problems and achieve higher success rates. Figure 5.1 shows planning times as a function of the number of hypothesis poses ($|H|$) in the start belief, i.e., start state uncertainty. The computational cost of computing transitions for a belief state is proportional to the number of hypotheses in it. As the start state uncertainty increases, the number of belief states with a high number of hypothesis poses encountered during the search also increases. As a result, the speedups offered by the lazy planners grow with the start state uncertainty.

5.4.2 Indoor navigation

This is a 3D navigation domain, where the robot state is defined by the tuple (x, y, θ). The map is known but the state of the robot is not directly observable, instead, the robot relies on a 1D LiDAR sensor mounted on it for perception. Computing expected observations from a given state require expensive raycasting operations to simulate the sensor. The belief over the robot's state is represented as a set of particles $H = \{h_1, h_2...\}$, each corresponding to a hypothesis robot pose h_i. Consequently, computing expected observations from a belief state requires performing raycasting from all hypothesis poses in the belief, making the belief transition computations expensive. The discrete action set $\mathcal{A}$ consists of predefined motion primitives. We evaluate two variants of the problem. For both variants we use 3 indoor environments similar to the one we see in Fig 5.3.

With stochastic transition dynamics

In this variant, uncertainty in the robot's state arises from stochastic transitions. Certain regions in the map induce probabilistic motions, where the robot reaches the intended end state with probability 0.5, while slipping left or right with equal likelihood. The robot's start state is known, and the objective is to reach a goal region G. The belief state is represented as a set of weighted particles, with the weight corresponding to the probability of each robot pose. Heuristic is the expected distance of the states in the belief from G, $heur(b) = \mathbb{E}_{s\sim b}\ \text{Dist}(s, G)$. Dist is the minimum

Planner	Success Rate	Plan Time	Cost
RTDP-Bel	0.81	153.04	14.14
Lazy RTDP-Bel ($\hat{Q}_{init}$)	0.90	32.46	14.26
Lazy RTDP-Bel ($\hat{Q}_{init}^{MDP}$)	0.90	38.76	14.14
LAO*	0.71	185.37	14.14
Lazy LAO* ($\hat{Q}_{init}$)	0.77	33.54	14.24
Lazy LAO* ($\hat{Q}_{init}^{MDP}$)	0.77	40.43	14.14

Table 5.2: Performance comparison on the indoor navigation with stochastic transition dynamics problem.

distance of s from G on the MDP defined by the transition model $\mathcal{T}$, making the heuristic admissible. Alternatively, Dist can be the Euclidean distance between s and G, providing a quick but inadmissible heuristic. We evaluate with the subsampled estimator, and since the computational expense of the belief transitions stems from the observation model, we also evaluate with the $\hat{Q}_{init}^{MDP}$ estimator. Results presented in Table 5.2 are averaged over 200 runs, with random start states and goal regions. The timeout was 300 seconds. We observed that the lazy planners were particularly effective in this problem, improving planning speeds by 5x while computing policies with a marginally higher expected cost. The $\hat{Q}_{init}^{MDP}$ estimator resulted in slightly higher planning times in comparison to the subsampling estimator, however, it is guaranteed to be a conservative estimate (while the subsampled estimator is not), resulting in solution costs identical to that of the vanilla planners.

With start state uncertainty

Planner	Success Rate	Planning Time	Cost
Goal Directed Navigation			
RTDP-Bel	0.73	139.32	19.57
Lazy RTDP-Bel	0.89	84.91	19.61
LAO*	0.87	96.47	19.57
Lazy LAO*	0.99	53.42	19.60
Information Gathering			
RTDP-Bel	1.0	39.17	14.39
Lazy RTDP-Bel	1.0	21.32	14.25
LAO*	1.0	32.21	14.42
Lazy LAO*	1.0	16.75	14.87

Table 5.3: Performance comparison on the indoor navigation with start state uncertainty problem.

In this variant, uncertainty in the robot's state arises from start state uncertainty, defined by an initial set of equally likely hypothesis poses. The transition dynamics is deterministic, allowing the belief to be represented as a set of unweighted particles. This problem has two versions: (1) the robot must localize itself, and (2) the robot must reach a goal region G. For the information-

gathering version, the goal belief, belief heuristic, and Q-estimator are defined identically to the manipulation problem. In the goal-directed version, they follow the definitions used for stochastic transition dynamics. Results for both versions, averaged over 200 runs, are presented in Table 5.3, with each run using a random set of 100 to 250 start hypothesis poses. Similar to the manipulation domain, the lazy planners on average took 60% of the planning time of their vanilla counterparts at the expense of marginally higher costs.

5.4.3 Rough terrain navigation

Planner	Success Rate	Planning Time	Cost
Goal Directed Navigation			
RTDP-Bel	0.01	-	-
Lazy RTDP-Bel	0.21	1297.33	12.66
FH Lazy RTDP-Bel	0.90	109.9	12.66
LAO*	0.09	-	-
Lazy LAO*	0.33	886.64	12.66
FH Lazy LAO*	0.90	139.52	12.66
Information Gathering			
RTDP-Bel	0.07	-	-
Lazy RTDP-Bel	0.19	1015.51	14.58
FH Lazy RTDP-Bel	0.83	110.38	14.58
LAO*	0.09	-	-
Lazy LAO*	0.26	642.78	14.58
FH Lazy LAO*	0.81	163.79	14.58

Table 5.4: Comparison on the outdoor navigation problem.

This is a 3D outdoor navigation problem with deterministic transitions and observations. Uncertainty in the robot's state arises from start state uncertainty, with the belief represented as a set of unweighted particles. Unlike indoor navigation, the robot lacks a 1D LiDAR sensor; instead, the environment contains landmarks that the robot can observe if within a predefined distance, making the observation model efficient to query. Consequently, both the transition and observation models are computationally cheap. However, in this domain, an action is feasible from a belief state only if it can be accurately tracked from all hypothesis states in the belief. Due to rough terrain, certain states may cause the robot to get stuck in pits, fail to climb inclines, or execute motions inaccurately. Hence, validating an action requires expensive rollouts in a physics simulator, making action verification the primary bottleneck. Since the expected trajectory that will be tracked from the hypothesis states if the action were valid is known, and the observation model is efficient, belief transitions can be computed quickly. The bottleneck operation is simply the verification of the validity of actions.

Similar to the indoor navigation case, we evaluate two versions of the problem. Table 5.4 presents the performance averaged over 100 runs, with each run containing 30–50 randomly sampled start

state hypothesis and a timeout of 2400 seconds. FH Lazy RTDP-Bel and FH Lazy LAO* correspond to the full-horizon lazy variants discussed in Section 5.3.2. We observe that the full-horizon lazy planners demonstrate strong performance boosting success rates by close to 90% in comparison to the vanilla counterparts. By deferring action evaluations until after policy computation, the full-horizon lazy planners achieve a 10x speed up over the 1-step lazy planners. The high computational expense of the physics simulations make the impact of laziness pronounced in this domain, highlighting the utility of lazy planners.

5.5 Conclusion

In this work, we introduced Lazy RTDP-Bel and Lazy LAO*, two specific instances of a broader class of lazy heuristic search solvers for POMDPs with expensive-to-compute belief state transitions. These algorithms leverage Q-value estimates to defer expensive belief transition evaluations until they are truly necessary. By postponing these computations, our approach significantly enhances planning efficiency while preserving solution quality. We validated our methods across three challenging robotics problems, achieving substantial speedups over conventional solvers in all of them.

III

Workspace Occupancy Uncertainty

6 A FRAMEWORK FOR MANIPULATING IN UNCERTAIN ENVIRONMENTS USING CONTACTS

6.1 Introduction

In the previous chapters, we developed planning frameworks that leverage contact feedback to reason about the pose of objects of interest, with the goal of reducing pose uncertainty sufficiently to complete object-centric manipulation tasks. In this chapter, we shift focus to a broader class of manipulation problems in which robots must operate in cluttered, partially known environments where visual sensing is unreliable or unavailable. Rather than reasoning solely about the pose of a specific target object, the robot must now reason about the structure of the surrounding workspace itself. Using contact feedback, the robot incrementally infers free space, obstacles, and feasible motion paths while executing the task, enabling robust manipulation in environments that cannot be reliably perceived through vision alone.

Robots frequently encounter such scenarios in real-world manipulation. Consider the scenario in Fig. 6.1, where a robot is tasked with reaching a shutoff valve inside a cabinet under a kitchen sink. Occlusions, inconvenient positioning, and lack of direct line-of-sight make it difficult to visually inspect the region around the valve. Much like humans, the robot must instead rely on contact feedback to maneuver around obstacles and reach its target. As it moves through the cavity, it inevitably collides with pipes or brackets. By reasoning about where these contacts occur and combining this information with prior knowledge of typical obstacle structures encountered in the environment—for example, pipes are commonly found in this space and they often span the entire width of the cabinet—the robot can iteratively refine its understanding of free space and adjust its trajectory to the goal (Fig. 6.2).

This example illustrates a broader class of manipulation problems where robots must leverage contacts to operate in unknown environments: retrieving an item from the back of a cluttered shelf, locating a switch hidden behind clutter, or reaching through scaffolding on a construction site. In such domains, strong priors about typical obstacle structures exist, but the specific obstacles in any given scene—and their precise placements—remain uncertain. To this end, we propose a theoretically complete and empirically efficient framework that integrates contact feedback with structural priors

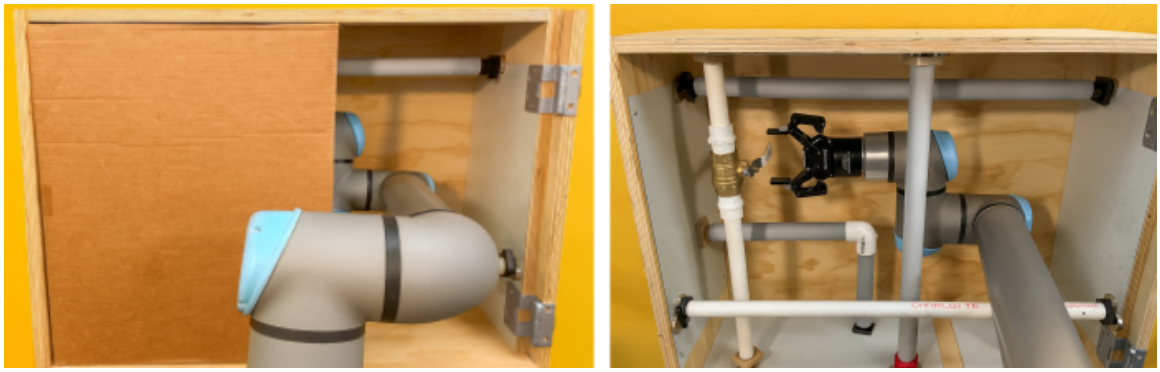

Figure 6.1: A robot reaching a shutoff valve at the back of a cluttered cabinet. As visual sensing is occluded by the shelf door (left), the robot relies on contact feedback to navigate around the obstacles (right).

to enable robust manipulation in uncertain environments. While prior chapters exploited contact to reduce uncertainty about a known object, the challenge here is fundamentally different. The robot must reason about an unknown workspace while simultaneously planning and executing motions through it.

Unlike much of the prior work on contact-rich manipulation, which has focused on high-resolution tactile sensing at the end-effector [133] [134], our approach addresses the fact that contacts in cluttered settings can occur anywhere along the arm. Full-body tactile skins [135] that provide such coverage remain costly and fragile, whereas joint positions and torques are widely available on modern manipulators and offer a noisy but accessible alternative. We therefore leverage torque-based contact feedback as the primary sensing modality, augmented by learned predictors that extrapolate the likely structure of the workspace from the sparse contact observations. The framework is carefully designed to handle the fact that both contact localization from joint torques and workspace prediction from sparse inputs are inherently noisy and prone to spurious hypotheses. At its core, the framework is an iterative planning and execution loop composed of three tightly coupled components:

1. **Contact Detection and Isolation Module:** We employ a momentum-based observer [45] to estimate external torques acting on the robot which allows robust detection of contact events anywhere along the manipulator. Once a contact is detected, a contact particle filter [136] estimates its likely location on the robot's surface. While binary detection is reliable, localization can be noisy, and our framework explicitly accounts for this uncertainty.

2. **Occupancy Estimation Module:** From the robot's interaction history, we construct a partial estimate of the workspace: regions at estimated contact locations are marked occupied, while regions swept without collision are marked free. Because this estimate only covers explored areas, we introduce two learned predictors—a CNN and a diffusion model—that extrapolate into unexplored regions by leveraging structural priors (e.g., pipes in a sink typically span the full width). By fusing such priors with contact history, the module predicts likely occupied and free regions beyond explored space, enabling the robot to avoid potential obstacles and reach the goal efficiently.

3. **Planning Module:** We integrate the workspace estimate with two existing frameworks: (i) Collision Hypothesis Sets (CHS) [36] and (ii) CMAX [137]. This integration biases the search toward paths most likely to succeed, while preserving robustness by ensuring that feasible solutions are never discarded due to incorrect contact localization or occupancy prediction.

In summary, the contribution of this work is the complete framework that unifies joint torque–based contact detection and localization, occupancy prediction, and planning into a single iterative loop. We evaluate the framework in both simulation and in the real world on a UR10e manipulator across two practical domestic tasks: (i) manipulating a valve under a kitchen sink surrounded by pipes, and (ii) retrieving a target object from a cluttered shelf. Results show that the framework reliably solves these tasks, achieving up to a 2× reduction in task completion time compared to baselines, with ablations confirming the contribution of each module.

6.2 Related Work

Joint torque sensing has long been used for contact detection and localization by comparing measured and model-predicted torques, with the residual representing external torques due to contact [138]. Momentum observers remain the standard for computing these residuals, valued for their robustness and low computational cost [45]. Contact localization is commonly performed with particle filters that maintain hypotheses over candidate surface points and update their likelihoods based on the observed residuals [136, 139]. Our framework directly leverages these ideas for contact detection and localization. However, localization from joint torques is inherently noisy: multiple contact points can explain the same residuals, and estimates are sensitive to sensor noise and model inaccuracies [140]. To address this, data-driven approaches have also been explored, such as treating localization as classification over discretized surface points [141] or inferring contacts directly from joint kinematics without torque sensing [142]. Such methods could serve as drop-in replacements or complements to our detection and localization module.

Unlike vision, which is information-rich, contact sensing provides extremely sparse and local feedback, making occupancy extrapolation from structural priors essential for efficient operation. The vision community has demonstrated the power of learned priors for inferring plausible content from incomplete observations—most notably in image inpainting [143, 144, 145, 146] and in 3D shape completion from partial scans [147, 148, 149, 150]. At the scene level, voxel-based semantic scene completion models [151, 152, 153] predict dense 3D occupancy from single-view input by learning a prior over likely structure, while point-cloud–based semantic scene completion networks such as S3CNet [154] and SCPNet [155] perform completion directly from sparse LiDAR point clouds. Our occupancy prediction module can be viewed as extending these ideas to the contact-sensing realm. To this end, we develop a CNN-based and a Diffusion-based predictor that is inspired from the original U-Net-style architecture [156], the masked-resampling strategy of RePaint [143], and the explicit mask conditioning idea used in the inpainting variants of Stable Diffusion [145, 157].

A number of planning frameworks for leveraging contacts in manipulation tasks have been studied. For example, [14, 17] formulate the problem of localizing objects of interest through contacts

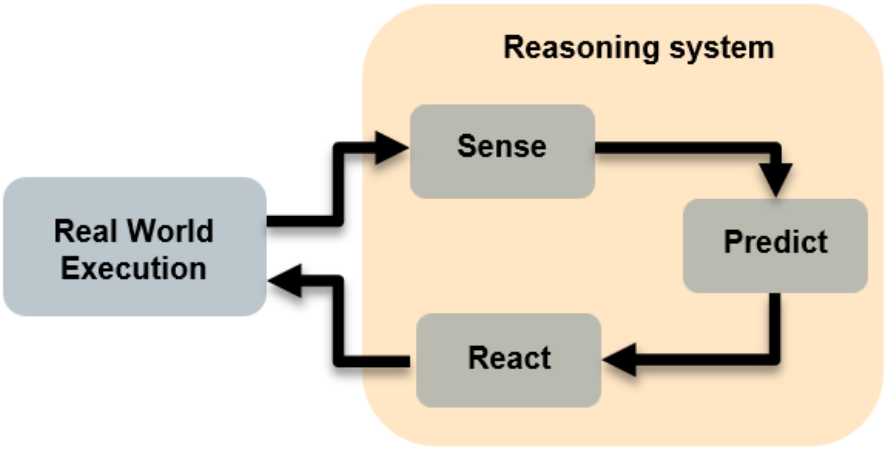

Figure 6.2: The overall reasoning system can be decomposed into three simple modules. As the manipulator navigates an unknown environment, the sensing module detects contacts with the environment and, upon contact, infers where along the arm the contact occurs. The history of contact observations is combined with prior knowledge about the environment to construct a mental model of the workspace via an occupancy prediction or estimation module. Using this evolving understanding of free and occupied space, the reaction module adjusts the robot's trajectory toward the goal. This process is repeated in a closed-loop planning and execution cycle.

as a POMDP, enabling active information gathering and robust task execution under uncertainty. [51] utilizes a greedy formulation for similar tasks. [158] proposes a hierarchical planning framework for leveraging contact detections from a simulated tactile skin, while [159] present an MPC-based framework for navigating through cluttered environments consisting of plants and tree trunks. Particularly relevant to our setting are (i) the Collision Hypothesis Set (CHS) representation [36, 37], developed for planning with reliable binary contact signals, and (ii) CMAX [137], a framework for planning under inaccurate models. While both are theoretically strong, they do not explicitly model or reason about the workspace, limiting their ability to exploit structural priors and often leading to repeated collisions in cluttered environments, as demonstrated by our results in Section 6.5. In this work, we augment these frameworks with our workspace estimate, yielding a theoretically sound yet empirically efficient approach.

6.3 Problem Setup

Let $\mathcal{Q}$ denote the configuration space of the manipulator, and $\mathcal{W}$ be a discrete binary voxel-grid approximation of the workspace, where each voxel is either occupied by an environmental obstacle or free. Let $\mathcal{W}_O \subset \mathcal{W}$ be the subset of occupied voxels. The mapping $\mathcal{R} : \mathcal{Q} \to 2^{\mathcal{W}}$ returns the set of workspace voxels occupied by the robot at configuration $q \in \mathcal{Q}$; a configuration is in collision if $\mathcal{R}(q) \cap \mathcal{W}_O \neq \emptyset$.

Following lattice-based approaches [160], we cast planning as search on a graph $G = (V, E)$, where vertices $V \subset \mathcal{Q}$ are discretized configurations and edges E are motion primitives from a discrete action set $\mathcal{A}$. In our case, $\mathcal{A}$ consists of linearly interpolated unit motions along each joint

dimension, with transition cost $c(q, q') = \|q' - q\|$. A path $\pi = (q_0, \ldots, q_T)$ is a sequence of such edges. Executing an edge $(q_{i-1}, q_i]$ either completes—certifying the swept volume $S(q_{i-1}, q_i) = \bigcup_{s\in(0,1]} \mathcal{R}((1-s)q_{i-1} + s\, q_i)$ as free—or detects first contact, in which case execution halts and the robot retracts to q_{i-1}. We define the execution operator $\mathcal{M}(q, \pi) = (q', c)$, which returns the configuration q' reached by attempting π from q and the cumulative execution cost incurred c.

At the start of execution, the occupancy grid $\mathcal{W}$—and hence $\mathcal{W}_O$—is only partially known; let the unobserved region be $\mathcal{W}_{\text{unknown}} \subset \mathcal{W}$. Proprioceptive observations $(q, \dot{q}, \tau)$ consisting of joint positions, velocities, and torques yield a reliable binary contact detection and a noisy estimate of the contact location. Contact-free traversals certify their swept volumes as free, whereas a contact indicates that $\mathcal{W}_O$ was intersected at some configuration q_{col} while executing the edge (q_i, q_{i+1}). The robot maintains a probabilistic occupancy grid $\hat{\mathcal{W}}_{\text{unknown}} : \mathcal{W} \to [0, 1]$, where each voxel encodes its probability of being occupied as an estimate of $\mathcal{W}_{\text{unknown}}$. This estimate is refined online from contact observations and extrapolated into unexplored regions using structural priors.

Given start and goal configurations $q_{\text{start}}, q_{\text{goal}} \in V$, the task is to reach q_{goal} from q_{start} despite partial knowledge of $\mathcal{W}_O$. Since collisions with unknown obstacles are unavoidable, the robot iteratively detects contacts, retracts, updates $\hat{\mathcal{W}}_{\text{unknown}}$, and replans. Within our iterative planning–execution framework, the problem can be stated as

$$\min_{\{\pi_k\}_{k=0}^{N-1}} \quad \sum_{k=0}^{N-1} c_k \quad \text{s.t.} \quad q_0 = q_{\text{start}}, \quad q_N = q_{\text{goal}}, \quad (q_{k+1}, c_k) = \mathcal{M}(q_k, \pi_k). \tag{6.1}$$

Here, π_k denotes the path computed at the k-th iteration of execution. The search problem is thus defined on G, with edge outcomes revealed through execution. Completeness with respect to G requires that if a collision-free path exists from q_{start} to q_{goal}, the planner eventually discovers it; if none exists, infeasibility is reported in finite time. Binary contact detections are treated as reliable, while contact localizations and occupancy predictions can be noisy.

6.4 Methodology

As outlined earlier, our framework follows an iterative planning–execution loop that can be decomposed into three modules. The contact detection and localization module uses proprioceptive feedback to detect and localize contacts, while the occupancy estimation module aggregates the history of such feedback into an evolving estimate of the workspace $\hat{\mathcal{W}}_{\text{unknown}}$ and extrapolates into unexplored regions using structural priors. The planning module then carefully reasons about the reliable binary contact detections and the estimated workspace $\hat{\mathcal{W}}_{\text{unknown}}$ to generate paths that are most likely to succeed while preserving completeness.

6.4.1 Contact Detection and Localization Module

We utilize the stream of joint configurations, velocities, and torques obtained during execution to i) detect contacts and ii) localize the contact on the robot manipulator. For this purpose, we leverage

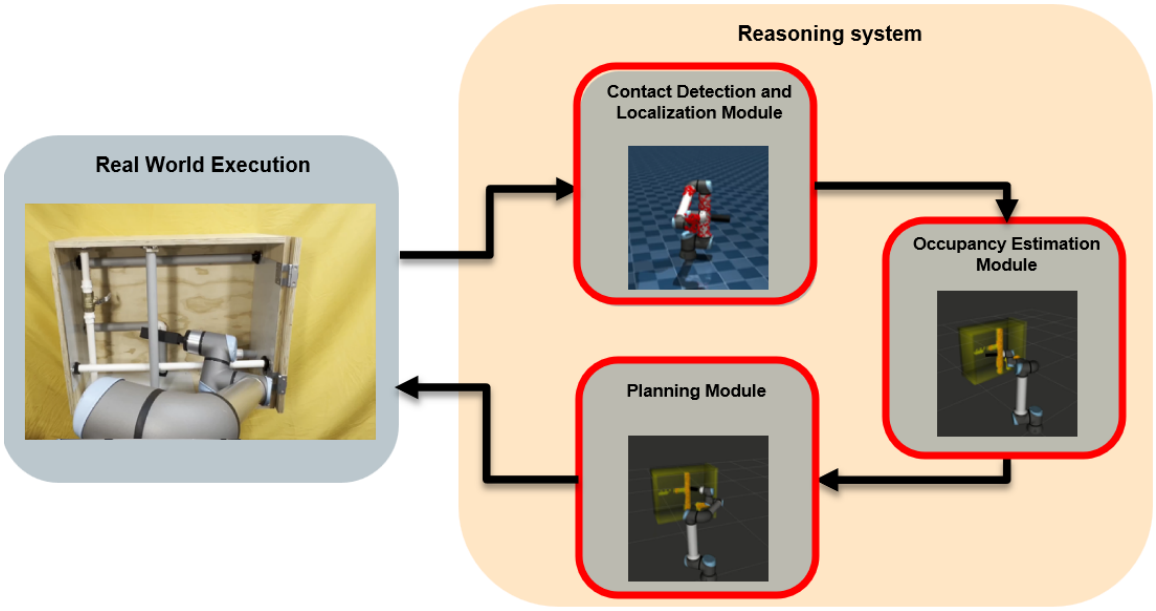

Figure 6.3: An illustration of the overall closed-loop framework. The reasoning system contains a contact detection and localization module that identifies the contact location. The history of proprioceptive feedback is used to construct a partial occupancy map of the workspace, which is then provided to the occupancy prediction module to infer a complete occupancy map. This map is then used by the planning module to compute a new path that is likely to succeed. The resulting plan is executed on the real robot until either a collision occurs or the goal is reached.

established techniques that combine external momentum observers with particle filters [136].

External Momentum Observer for Detection

Following [45], the generalized momentum is defined as $p(t) = H(q)v$, where $H(q)$ is the joint-space inertia matrix and v the generalized velocity. From the manipulator dynamics, its time derivative is given by $\dot{p} = \tau + \tau_{\text{ext}} + C(q,v)^\top v - g(q)$, with τ the commanded joint torques, $C(q,v)$ the Coriolis and centrifugal terms, $g(q)$ gravity, and τ_{ext} the external joint torques due to contact.

The momentum observer then defines a residual signal between the measured momentum and the momentum predicted by the manipulator dynamics model:

$$r(t) = K_O\left(p(t) - \int_0^t \left(\tau + C(q,v)^\top v - g(q) + r(s)\right) ds\right),$$

where $K_O > 0$ is a diagonal gain matrix. This residual evolves as $\dot{r}(t) = K_O(\tau_{\text{ext}} - r(t))$. So each component of r acts as a low-pass filtered estimate of the external torque, making the residual a good approximation of τ_{ext}. In nominal free-motion, r remains near zero up to modeling errors and sensor noise. When an unexpected contact occurs, the external torque contribution drives the residual away from zero. A collision is declared when $|r(t)|$ exceeds a tuned threshold. The external momentum observer thus offers a lightweight, real-time method for reliable detection.

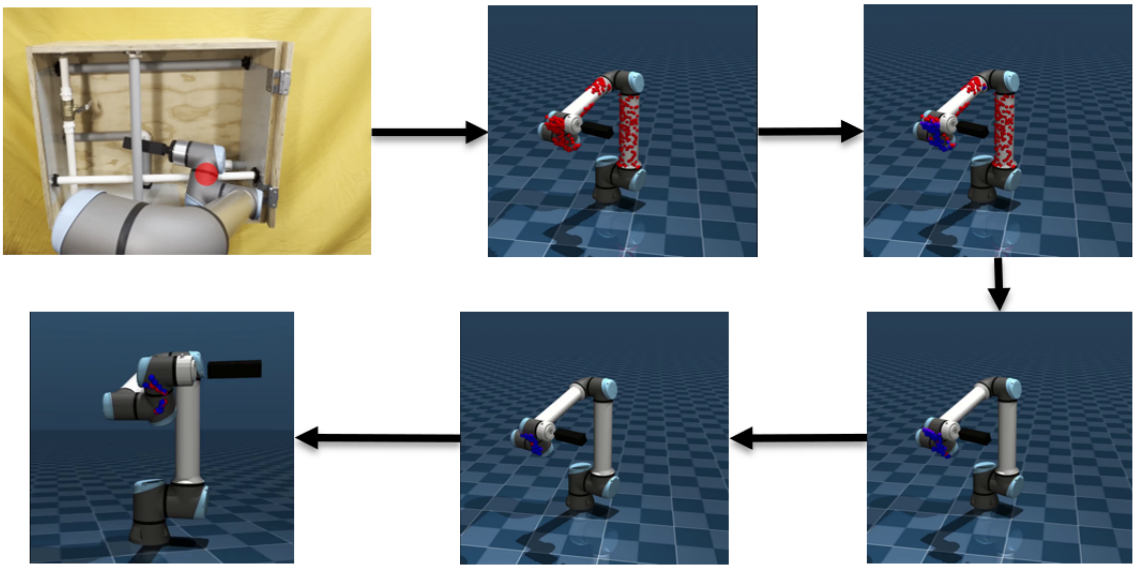

Figure 6.4: Example run of the contact particle filter. Top left: the robot makes contact with a pipe at the location highlighted in red. The remaining images show successive iterations of the particle filter. Particles transition across iterations as the belief over the contact location is refined, with the final image showing convergence to the ground truth contact point. Red particles denote the full particle set at the beginning of each iteration, while blue particles indicate the high likelihood particles identified during that iteration.

Contact Particle Filter for Localization

Once a contact has been detected, the next step is to localize its point of occurrence on the manipulator. For a given configuration q and candidate location x_c with Jacobian $J_{x_c}(q)$, the joint torque induced by a contact force $F_c \in \mathbb{R}^3$ is $J_{x_c}(q)^\top F_c$. The localization task can therefore be posed as finding the pair (x_c, F_c) that best explains the observed residual $r(t) \approx \tau_{\text{ext}}$. This joint optimization is non-convex due to the coupling of location and force variables. Following [136], we approximate the solution by sampling candidate contact locations on the robot surface and solving, for each, a convex quadratic program (QP) in F_c, whose cost measures how well the hypothesized location explains the observed external torques.

This idea is realized in the Contact Particle Filter (CPF) [136], which maintains a set of particles $X_t = \{x_t^{[1]}, \ldots, x_t^{[M]}\}$ representing hypotheses over possible contact locations. The particle set is initialized from the set of candidate surface points $\mathcal{S}$ (i.e., $X_0 \subseteq \mathcal{S}$), and at each step is updated through resampling, perturbation, and reweighting. The measurement model assigns a likelihood to each particle based on the QP formulation. For a particle $x_t^{[m]}$ with Jacobian $J_{x_t^{[m]}}(q)$, we solve

$$\ell(x_t^{[m]}) = \min_{F_c \in \mathcal{F}(x_t^{[m]})} \| r(t) - J_{x_t^{[m]}}(q)^\top F_c \|^2_{\Sigma_{\text{meas}}^{-1}},$$

where $\|z\|_A^2 = z^\top A z$, $\mathcal{F}(x_t^{[m]})$ is the friction cone at $x_t^{[m]}$, and Σ_{meas} encodes measurement noise. The optimal cost is then converted into a likelihood as $p(r(t) \mid x_t^{[m]}) \propto \exp(-\frac{1}{2}\,\ell(x_t^{[m]}))$. Particles that yield smaller residual errors are assigned higher likelihoods, and resampling concentrates the

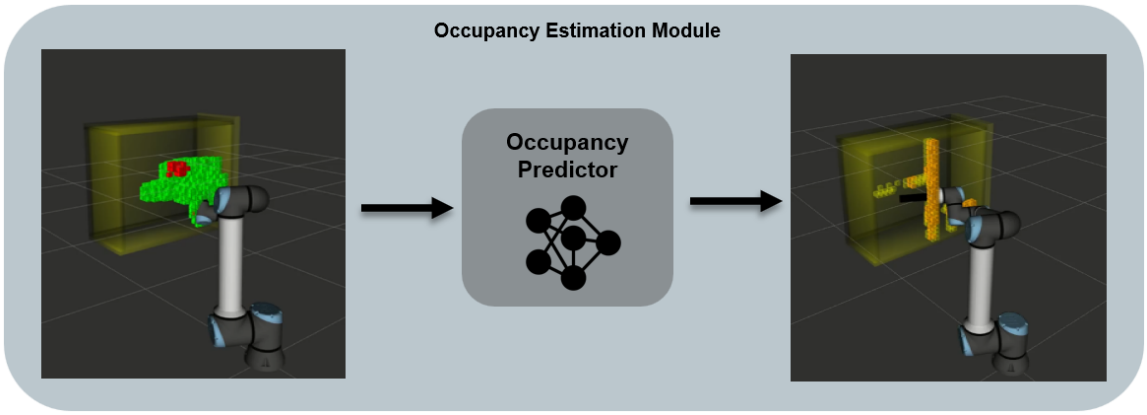

Figure 6.5: An occupancy predictor takes a partial occupancy map of the workspace, constructed from the history of proprioceptive observations, and uses it to predict the complete workspace occupancy map.

distribution around regions of the surface consistent with the observed torques. In order to prevent particle impoverishment and stabilize the filter, a simple motion model is applied between updates, where each particle undergoes a Gaussian perturbation followed by projection back to the nearest valid point on the robot surface. After t iterations, the centroid of the largest surviving cluster is returned as the contact location estimate. An example run of the CPF can be seen in Fig. 6.4

Robustifying Contact Localization

As noted in prior work [140], multiple pairs of contact locations and forces can yield similar torque residuals, making the CPF susceptible to spurious hypotheses. To improve robustness, we incorporate a few additional refinements. First, we restrict the set of candidate points $\mathcal{S}$ to those whose motion is physically consistent with contact. Each $x \in \mathcal{S}$ has an outward surface normal $n(x)$ and an induced Cartesian velocity $\dot{x}(q, \dot{q})$. A contact can only occur if the point is moving into the environment rather than away from it, which requires $\langle n(x), \dot{x}(q, \dot{q}) \rangle > 0$. The set of kinematically admissible candidates is therefore $\mathcal{S}_{\text{active}}(q, \dot{q}) = \{x \in \mathcal{S} \mid \langle n(x), \dot{x}(q, \dot{q}) \rangle > 0\}$. Second, we enforce workspace consistency by requiring that possible contact points also lie within unexplored regions of the workspace, i.e., $\mathcal{S}_{\text{active}} \cap \hat{\mathcal{W}}_{\text{unknown}}$. Finally, we also incorporate a learned link estimator: a lightweight MLP processes a temporal window of external torques τ_{ext} to predict the probability of the contact having occurred at each link. These probabilities are used to lightly reweight the particles, biasing the filter toward hypotheses supported by both dynamics and the learned estimator.

6.4.2 Occupancy Prediction

The history of proprioceptive observations provides a direct but sparse estimate of the unknown workspace $\hat{\mathcal{W}}_{\text{unknown}}$. Whenever an action is executed without contact, all voxels swept by the manipulator are certified as free. Conversely, voxels corresponding to estimated contact locations are assigned high probability of occupancy. This produces a consistent map, but one that is highly in-

complete: only regions the robot has explicitly interacted with are labeled. In cluttered settings such as sink cabinets or shelves, this limitation can lead to repeated collisions and inefficient exploration.

To mitigate this sparsity, we leverage structural priors about common obstacle geometries to extrapolate occupancy into unobserved regions. We cast this as a prediction problem: given a partially observed voxel grid, infer the occupancy probabilities of the unobserved voxels. The resulting prediction serves as a richer estimate of $\hat{\mathcal{W}}_{\text{unknown}}$ and guides subsequent planning (Fig. 6.5).

We study two learned predictors for this task: a convolutional model and a generative diffusion model. Both take as input a voxel grid $\mathbf{x} \in \{0, 0.5, 1\}^{H \times W \times D}$, matching the dimensions of $\hat{\mathcal{W}}_{\text{unknown}}$, where 0 denotes known free space, 1 denotes known occupied space, and 0.5 denotes unknown. Each predictor outputs a grid of the same dimensions, assigning to every voxel (i, j, k) a continuous occupancy probability $\hat{o}_{ijk} \in [0, 1]$.

CNN Predictor

Building on U-Net's demonstrated effectiveness in vision tasks, our first predictor is a 3D U-Net [156] adapted to sparse contact observations. The network follows the standard encoder–decoder design with skip connections: partial observations are processed through successive downsampling and upsampling blocks, with features passed across stages to retain spatial detail. To better capture long-range structure from limited evidence, the encoder employs large kernels with aggressive downsampling. After each encoder block, we insert a *deformable sampling layer* to further enrich feature extraction.

The goal of this layer is to let the network adaptively sample data-dependent features beyond the fixed voxel grid. In standard 3D deformable attention [161], the feature at each voxel $\mathbf{g}$ is fed into prediction heads that output a set of sampling offsets $\{\Delta\mathbf{g}_p\}_{p=1}^{n_{\text{points}}}$ and corresponding scores $\{s_p\}_{p=1}^{n_{\text{points}}}$, producing the aggregated output feature: $\mathbf{y}(\mathbf{g}) = \sum_{p=1}^{n_{\text{points}}} \text{softmax}_p\left(s_p\right) \mathbf{v}\Big(\mathbf{g} + \Delta\mathbf{g}_p\Big)$, where $\mathbf{v}(\cdot)$ denotes the feature map from which values are sampled. However, with extremely sparse contact inputs, the softmax weights and scores can become unstable and amplify noise. To improve robustness, we keep the learnable offsets but replace the weighted sum with a max operation:

$$\mathbf{y}(\mathbf{g}) = \max_{p=1,\ldots,n_{\text{points}}} \mathbf{v}\Big(\mathbf{g} + \Delta\mathbf{g}_p\Big), \tag{6.2}$$

which preserves adaptive sampling behavior while avoiding noisy aggregation. The decoder mirrors the encoder and outputs voxel-wise occupancy probabilities $\hat{o} \in [0, 1]$ through a final sigmoid layer, which we interpret as $\hat{\mathcal{W}}_{\text{unknown}}$. Training uses standard binary cross-entropy loss augmented with a per-voxel binary entropy regularizer to discourage over-confident predictions.

Diffusion Predictor

To capture the multi-modality inherent in extrapolating from sparse contacts, we also develop a generative predictor based on diffusion models. We adapt the same U-Net backbone as above, conditioning on the partial observation $\mathbf{x}^{\text{obs}}$ and a binary mask m marking observed ($m = 1$) and unobserved ($m = 0$) voxels. For inference, we apply DPM-Solver++ [162], a high-order ODE sampler

that enables fast and stable generation.

A key challenge is generating occupancy grids consistent with the observed regions provided as part of the input. Standard diffusion inpainting gradually denoises the entire grid, which can drift even in known voxels. To prevent this, we adopt a *known-region reinforcement* strategy inspired by RePaint [143]: after each denoising step, the sample $\mathbf{x}_t$ is updated as $\mathbf{x}_t \leftarrow m \odot \mathbf{x}_t^{\text{obs}} + (1-m) \odot \mathbf{x}_t$, where $\mathbf{x}_t^{\text{obs}}$ is the noisy version of the observed input. This reinsertion biases the predictions to be consistent with the partial inputs.

To further improve controllability, we incorporate *classifier-free guidance* [145]. Given a conditional noise prediction $\epsilon_\theta(\mathbf{x}_t, t, \mathbf{x}^{\text{obs}}, \mathbf{m})$ and its unconditional counterpart $\epsilon_\theta(\mathbf{x}_t, t, \varnothing)$, the update is

$$\hat{\epsilon}_\theta = \epsilon_\theta(\mathbf{x}_t, t, \varnothing, \mathbf{m}) + w\Big(\epsilon_\theta(\mathbf{x}_t, t, \mathbf{x}^{\text{obs}}, \mathbf{m}) - \epsilon_\theta(\mathbf{x}_t, t, \varnothing, \mathbf{m})\Big),$$

where $w > 1$ controls the strength of conditioning. This balances two objectives: enforcing consistency with observed voxels while still exploring diverse and plausible completions in unobserved regions. Unlike the deterministic CNN predictor, the diffusion model represents the full conditional distribution $p(\mathbf{x} \mid \mathbf{x}^{\text{obs}})$. In practice, we generate a batch of samples and take their voxel-wise mean, yielding an expected occupancy estimate that we interpret as $\hat{\mathcal{W}}_{\text{unknown}}$.

6.4.3 Planning Module

Given the observation history and the estimated occupancy $\hat{\mathcal{W}}_{\text{unknown}}$, the planning module computes paths that maximize success probability—avoiding collisions while reaching the goal with minimum execution cost. To guarantee completeness, it must avoid pruning feasible solutions due to noisy predictions while preventing indefinite execution of failing plans. To achieve this, we propose two variants that combine the estimated occupancy map with two previously developed ideas: (i) *Collision Hypothesis Sets* (CHS) [36], which encode reliable binary contact detections into a sparse set-based representation; and (ii) *CMAX* [137], a framework for planning with inaccurate world models.

Integration with CHS

A single CHS $\kappa_i \subset \mathcal{W}$ is the set of voxels that could explain a collision observed when executing an edge $e = (q_i, q_{i+1})$. If a collision is detected at configuration q_{col}, then among the voxels intersecting the active robot surface, at least one must be occupied (refer Fig. 6.6); this set of potentially occupied voxels is recorded as κ_i. The CHS representation maintains the collection $\kappa = \{\kappa_1, \kappa_2, \ldots\}$ of all such hypotheses accumulated during execution.

Given a hypothesis κ_i, the probability that an edge e is in collision is defined in proportion to the fraction of the hypothesis set swept by the edge:

$$P(v(e) = 0 \mid \kappa_i) = \frac{|S(e) \cap \kappa_i|}{|\kappa_i|}. \tag{6.3}$$

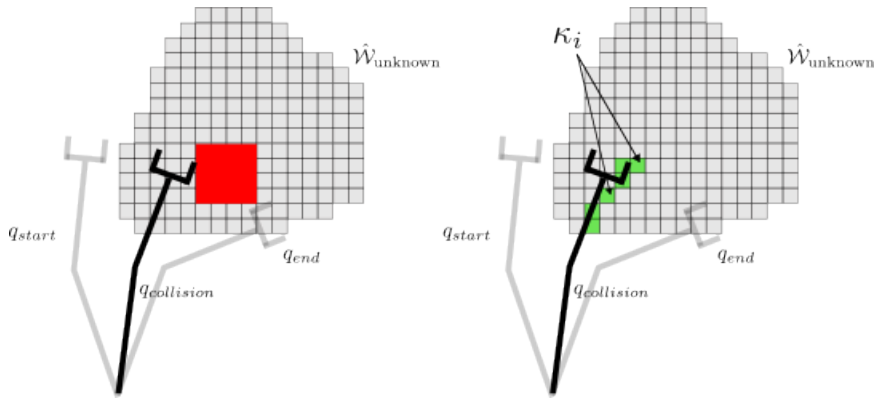

Figure 6.6: The collision hypothesis set κ_i (in green) corresponds to the voxels intersecting with the active surface of the robot at the time of collision.

If $S(e)$ fully contains κ_i, then $P(v(e) = 0 \mid \kappa_i) = 1$, since by construction κ_i must include at least one occupied voxel. Assuming independence across hypotheses, the probability that e is valid given all CHSs is then given by:

$$P(v(e) = 1 \mid \kappa) = \prod_i \Big(1 - P(v(e) = 0 \mid \kappa_i)\Big). \tag{6.4}$$

Edges that have already been attempted and invalidated receive zero validity probability, ensuring they are never re-executed, while other edges are never prematurely ruled out.

While CHS is conservative and accurate, it does not explicitly construct the workspace, which limits the ability to exploit structural priors, often resulting in many contacts and poor performance in cluttered regions. We therefore augment it with $\hat{\mathcal{W}}_{\text{unknown}}$, biasing search toward safer trajectories. Concretely, we perform a search on the graph $\mathcal{G}$ with a modified edge cost function. Each edge cost is augmented with penalties derived from both CHS and the occupancy map, reflecting the likelihood that the edge leads to a collision under each representation. Edges assigned infinite cost are deemed infeasible and never selected for execution. Formally, for each edge e, the cost is defined as

$$c(e) = c_{\text{dist}}(e) + \alpha\, c_{\text{CHS}}(e) + \beta\, c_{\hat{\mathcal{W}}_{\text{unknown}}}(e), \tag{6.5}$$

where $c_{\text{dist}}(e)$ is the configuration-space distance, $c_{\text{CHS}}(e) \propto \frac{1}{P(x(e)=1|\kappa)}$ penalizes edges with low CHS validity, and $c_{\hat{\mathcal{W}}_{\text{unknown}}}(e)$ penalizes edges likely to collide according to the occupancy estimate. In practice, $\alpha \gg \beta$ as CHS is sparse but highly reliable and can prune edges outright (edges with zero validity probability according to Eqn. 6.4 incur infinite cost), whereas occupancy-based penalties are bounded and used only to bias the search.

Integration with CMAX

CMAX [137] addresses planning with inaccurate models by interleaving planning and execution while preserving real-time operation and completeness. When execution reveals that a transition differs from the nominal model, CMAX does not repair the model but instead biases the planner away

from such transitions by inflating their cost—typically by placing hyperspheres around discovered discrepancies so that nearby state–action pairs incur additional penalty. As new discrepancies are observed, the process repeats, ensuring the planner ultimately avoids all incorrect transitions and still reaches the goal.

In our setting, the nominal model assumes the workspace is empty, so collisions reveal discrepancies between $\hat{f}$ and the true dynamics f. Let $\chi \subseteq S \times A$ denote the set of state–action pairs producing such discrepancies. The CMAX penalty for a candidate (s, a) is defined as

$$c_{\mathrm{CMAX}}(s,a) \;\propto \max_{\substack{(s',a')\in\chi,\, a'=a \\ d(s,s')<\delta}} \frac{1}{d(s,s')}, \tag{6.6}$$

where $d(\cdot,\cdot)$ is the Euclidean distance in configuration space and δ is a domain-dependent hypersphere radius. Transitions that repeat action a near a previously invalidated state s' incur large penalties, diverging as $d(s,s') \to 0$, while those outside all δ-neighborhoods remain unpenalized. Although theoretically sound, this framework by itself can be less effective in our setting, since penalization is applied in the state–action space. Consequently, the robot can execute different transitions that appear unrelated in state–action terms yet traverse overlapping workspace regions, resulting in repeated collisions.

To mitigate this limitation, we augment the CMAX penalty with workspace-level costs derived from our occupancy estimate, similar to the CHS case. Hence, similar to Eqn. 6.5, the cost of an edge is defined as the linear sum of the baseline path length, the CMAX penalties, and the occupancy-based penalties $c_{\hat{\mathcal{W}}_{\mathrm{unknown}}}(e)$. As in the CHS case, CMAX can prune edges outright by assigning infinite cost to known-invalid transitions, whereas occupancy-based penalties are bounded and serve only to bias the search. This coupling preserves the completeness guarantees of CMAX while adding workspace-aware reasoning that reduces repeated collisions in cluttered settings.

$$c(e) = c_{\mathrm{dist}}(e) + \alpha\, c_{\mathrm{CMAX}}(e) + \beta\, c_{\hat{\mathcal{W}}_{\mathrm{unknown}}}(e), \tag{6.7}$$

Completeness Discussion Both the CHS representation and the CMAX framework are theoretically sound and complete, guaranteeing recovery of a path in $\mathcal{G}$ (if one exists) when binary contact signals are used to identify infeasible edges. They operate through distinct mechanisms: CHS constructs a hypothesis set κ_i containing at least one truly occupied voxel and permanently invalidates edges whose swept volume fully covers κ_i, whereas CMAX records discrepancies and inflates the costs of nearby state–action pairs, assigning infinite cost only to the invalidated transition itself. In both cases, only edges that are provably infeasible are removed, while all other edges remain in the search space; hence, a feasible edge is never falsely discarded. Since the estimated workspace is used solely to bias costs and never to prune, all feasible edges in $\mathcal{G}$ remain admissible, as guaranteed by the vanilla variants. Consequently, the search will eventually identify a feasible path if one exists—though it may first explore paths that appear promising under $\hat{\mathcal{W}}_{\mathrm{unknown}}$ and then backtrack when they are invalidated— establishing completeness with respect to $\mathcal{G}$.

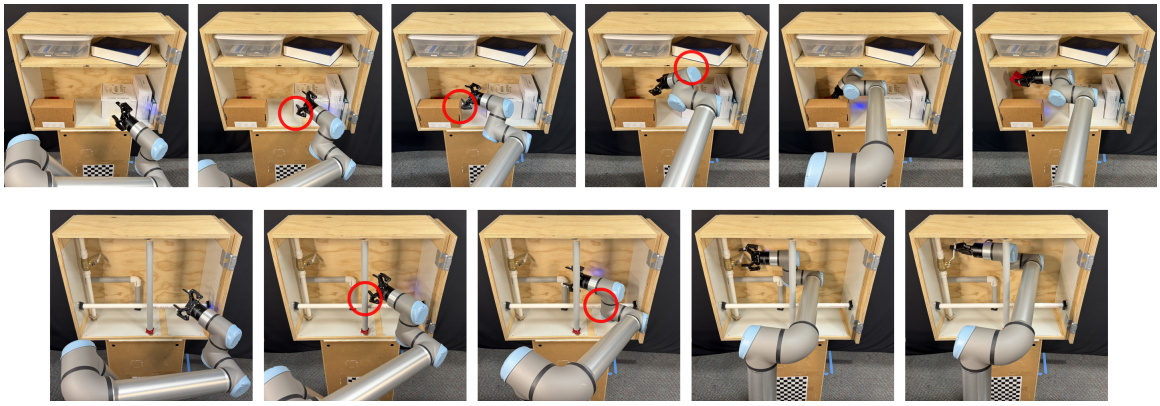

Figure 6.7: Example runs of the framework on the object retrieval task (top) and valve manipulation task (bottom). In both cases, the robot initially collides with environmental obstacles (highlighted in red) before identifying a feasible path that allows it to successfully reach the target.

6.5 Results

We evaluate the proposed framework using a UR10e manipulator in two representative domains that capture the challenges of manipulating in the blind. The first domain involves *valve manipulation under a sink cabinet*, where the robot must reach a shutoff valve mounted at the back wall while maneuvering around pipes that span across the cavity. This setting emphasizes strong structural priors, as pipes are slender and typically extend across the full width of the cabinet. The second domain involves *object retrieval from a cluttered bookshelf*, where the robot must reach deep into a shelf to retrieve an item placed at the back while interacting with books, shelf partitions, and other irregularly shaped objects (Fig. 6.7).

To illustrate the impact of the different modules, we evaluate the following planner variants. **CHS** and **CMAX** are the vanilla frameworks that do not utilize the estimated workspace occupancy $\hat{\mathcal{W}}_{\text{unknown}}$. **CHS + CNN/Diffusion** and **CMAX + CNN/Diffusion** are the proposed extensions that incorporate the estimated occupancy, obtained from either a CNN- or diffusion-based predictor. Finally, we consider variants that bypass both CHS and CMAX and instead plan directly on the estimated occupancy map $\hat{\mathcal{W}}_{\text{unknown}}$. In these cases, a voxel is treated as occupied if its estimated occupancy exceeds a fixed threshold (0.8), and any edge intersecting such voxels is ruled infeasible. Similar to Eqn. 6.5, edge costs here consist of configuration distance and occupancy penalties. These variants are denoted as $\hat{\mathcal{W}}_{\textbf{unknown}}$ **(no pred)**, and $\hat{\mathcal{W}}_{\textbf{unknown}}$ **+ CNN/Diffusion**, depending on which predictor they employ.

We evaluate the frameworks in both simulation and real-world across the two domains. All planners used Weighted A* search with a planning timeout of 5 s per iteration. A trial was declared a failure if (i) no solution was found within the timeout or (ii) the framework exceeded 20 plan–execute iterations without reaching the goal. In the reported tables, "Num iters" denotes the number of plan–execute iterations required to solve a problem, where each iteration consists of executing the planned motion, estimating contacts, updating $\hat{\mathcal{W}}_{\text{unknown}}$, and replanning. Planning, execution,

Table 6.1: Comparison of planners in simulation for valve manipulation task (pipes).

Planner	Success Rate	Num Iter.	Planning Time (s)	Execution Time (s)	Prediction Time (s)	Total Time (s)
CHS	0.83	11.15	0.48	346.66	0	347.14
CHS + CNN	**1.00**	5.76	2.41	195.97	1.84	200.22
CHS + Diffusion	0.97	4.95	1.52	191.94	97.64	291.10
CMAX	0.57	13.14	16.03	408.29	0	424.31
CMAX + CNN	0.86	6.14	11.48	237.82	1.96	251.26
CMAX + Diffusion	0.78	5.93	11.63	184.34	116.96	312.94
$\hat{W}_{\text{unknown}}$ (no pred)	0.59	10.95	0.13	333.57	0	333.70
$\hat{W}_{\text{unknown}}$ + CNN	0.82	5.86	1.51	192.02	1.87	**195.41**
$\hat{W}_{\text{unknown}}$ + Diffusion	0.79	5.04	1.46	187.82	99.30	288.59

occupancy prediction, and contact-estimation times are reported as cumulative time spent on each component across all iterations. The "Total time" entry corresponds to the overall time required by each variant to complete the task.

In both domains, the unknown portion of the workspace $\mathcal{W}_{\text{unknown}}$ corresponds to a tightly constrained cabinet with multiple potential obstacles. In the pipe domain, $\mathcal{W}_{\text{unknown}}$ contains 5–12 randomly placed pipe segments inside the cabinet, while in the shelf domain, $\mathcal{W}_{\text{unknown}}$ contains 1–4 shelf partitions and 6–12 randomly placed household objects. The contact estimation module was used only in the real-world studies; in simulation, contact estimation was modeled by perturbing the true collision point in Cartesian space with zero-mean Gaussian noise (standard deviation 3 cm independently along each coordinate) before projecting it back onto the robot surface as the estimated contact location.

Training Data for Occupancy Predictor Training data for the occupancy prediction modules were generated by sampling random actions and recording interactions with environmental obstacles in both domains. For each datapoint, a random scene was sampled, and 1–7 actions were executed. The input is the partial occupancy map derived from these sweeps. The output has voxels corresponding to objects the robot directly interacted with marked occupied, while all other voxels are labeled free. To simplify learning and reduce ambiguity, supervision is restricted to these interacted objects rather than the entire scene. This design choice reflects the limited spatial correlation between objects in our environments and yields a more tractable prediction problem.

6.5.1 Simulation Results

Tables 6.1 and 6.2 summarize the simulation results for both domains, averaged over 200 randomly generated problem instances. Overall, the **CHS + CNN** variant achieved the strongest performance, solving 100% of all sampled problems. This variant effectively combines the strengths of both modules: CNN-based predictions bias the search away from repeated collisions, while the CHS representation ensures robustness by falling back to reliable binary detections when predictions

Table 6.2: Comparison of planners in simulation for object retrieval task (shelf).

Planner	Success Rate	Num Iter.	Planning Time (s)	Execution Time (s)	Prediction Time (s)	Total Time (s)
CHS	0.52	10.69	12.55	309.47	0	322.02
CHS + CNN	**0.94**	4.20	11.22	135.56	1.55	**148.34**
CHS + Diffusion	0.91	4.15	13.39	141.98	80.10	240.53
CMAX	0.33	13.88	12.38	380.32	0	392.69
CMAX + CNN	0.73	4.11	9.91	141.47	1.56	152.95
CMAX + Diffusion	0.66	5.67	15.50	180.90	100.93	303.66
$\hat{W}_{\text{unknown}}$ (no pred)	0.47	12.83	18.54	346.62	0	365.16
$\hat{W}_{\text{unknown}}$ + CNN	0.89	5.08	14.20	168.53	1.42	184.15
$\hat{W}_{\text{unknown}}$ + Diffusion	0.81	4.81	16.32	159.97	89.34	271.25

are noisy. By contrast, **CHS alone** achieves lower success rates, as the robot often requires many more interactions with the environment before reaching the goal. On average, the total time required by CHS + CNN is nearly half that of CHS, confirming the empirical speedup provided by the occupancy estimation and prediction module. Similarly, planning directly with just the workspace estimate $\hat{\mathcal{W}}_{\text{unknown}}$ (without CHS or CMAX) yields lower success rates due to false occupancy estimations causing the planner to incorrectly declare infeasibility when a solution exists. Combining the two approaches, therefore provides the best of both worlds—leaning on the occupancy estimator when predictions are accurate, while falling back on binary detections when they are not.

Between the two base planners, CHS consistently outperforms CMAX. Although CHS does not explicitly build a workspace estimate, its hypothesis-set representation still captures portions of the workspace that are potentially occupied, enabling it to avoid redundant collisions. CMAX, by contrast, penalizes discrepancies only in state–action space and therefore suffers from repeated collisions with obstacles. When augmented with occupancy prediction, however, CMAX shows clear improvements: **CMAX + CNN** narrows the performance gap to CHS, demonstrating the utility of workspace-level reasoning. Across both base frameworks, adding a predictor reduces the number of plan–execute iterations by more than half, directly translating into lower execution times and faster task completion.

A domain-level comparison shows that shelf environments are more challenging: baseline planners without predictors (CHS, CMAX, and $\hat{\mathcal{W}}_{\text{unknown}}$) achieve lower success rates and higher planning times on shelves than on pipes due to heavier clutter and more complex maneuvers required. However, when combined with prediction, success rates in shelves match or even exceed those in pipes, since predictors can exploit the structured regularities of shelves more effectively than randomized pipe routings. On average, predictor-augmented frameworks reduce total task time by a factor of 1.5–2×, with improvement particularly pronounced in shelf domains where structural priors are strongest.

Finally, we note the trade-off between CNN and diffusion predictors. Both reduce iterations and execution time to a similar extent, confirming comparable prediction quality. However, diffusion-

Table 6.3: Real-world valve manipulation and object retrieval results

Planner	Num Iter.	Plan. (s)	Exec. (s)	Pred. (s)	Contact Est. (s)	Total (s)
Pipes (Valve Manipulation)						
$\hat{W}_{\text{unknown}}$	10.7	0.09	324.2	0	21.8	357.1
$\hat{W}_{\text{unknown}}$ + CNN	5.0	5.17	192.8	1.61	10.7	214.5
CHS	11.0	0.25	324.1	0	24.7	360.4
CHS + CNN	3.8	1.44	161.2	1.11	8.9	175.9
Shelf (Object Retrieval)						
$\hat{W}_{\text{unknown}}$	13.3	20.6	404.5	0	26.0	466.0
$\hat{W}_{\text{unknown}}$ + CNN	5.7	12.7	168.5	1.71	11.4	199.8
CHS	12.6	19.8	344.7	0	27.9	407.5
CHS + CNN	4.1	11.3	110.6	1.60	10.8	138.0

based predictors are computationally intensive: generating each occupancy update requires sampling (50 samples in our implementation) and voxelwise averaging, incurring an additional ∼20 s per prediction. In contrast, the CNN model provides fast inference (∼0.3s per prediction) while achieving equivalent planning and execution gains. As a result, CNN-based prediction offers the most practical balance between accuracy and efficiency, while diffusion provides a proof of concept for handling multi-modality at the cost of higher runtime.

6.5.2 Real Robot Results

We further validated the framework on a UR10e manipulator in the real world across both domains, using the best-performing variants from simulation. Results are averaged over 20 runs per domain. A key difference from simulation is the use of raw proprioceptive signals for contact estimation: joint currents (linearly proportional to joint torques) were monitored to estimate external torques τ_{ext}, and the contact particle filter was run with 250 particles per update.

The performance trends observed in simulation largely carried over to hardware (Table 6.3). In particular, the **CHS + CNN** variant consistently outperformed all others, reducing execution time to less than half of vanilla CHS. For example, in the shelf domain, CHS + CNN reduced the total task time from 407.5 s (CHS) to 138.0 s. Similarly, in the pipe domain, CHS + CNN completed tasks in 175.9 s compared to 360.4 s for CHS. These results demonstrate that the benefits of occupancy prediction are not limited to simulation but transfer effectively to real-world execution.

Assuming a localization error of under 4cm as successful, the contact detection and localization module achieved 73% accuracy with the optimizations from Section 6.4.1 (Robustifying CPF) enabled. An ablation study on a held-out dataset attributed a 20% accuracy gain to these optimizations compared to a baseline without them. Although localization remains noisy, the CHS abstraction ensures that binary detections provide a reliable fallback when predictions are incorrect.

The benefits of occupancy prediction were most pronounced in the shelf domain, where naive

CHS and $\hat{\mathcal{W}}_{\text{unknown}}$ variants often required repeated probing to escape occluded regions. By leveraging predicted occupancy, the robot was able to anticipate unseen obstacles and reduce redundant collisions. Importantly, the CHS representation provided robustness against prediction errors: when contact was mislocalized, CHS still guided the planner away from the true contact region, allowing the robot to progress without re-colliding. In contrast, the $\hat{\mathcal{W}}_{\text{unknown}}$ + CNN variant typically required one or two additional iterations to revisit the same obstacle before updating its estimate. This explains the performance gap between $\hat{\mathcal{W}}_{\text{unknown}}$ + CNN and CHS + CNN, despite both using the same occupancy predictor.

6.6 Future Work And Conclusion

A promising future direction is to augment occupancy prediction with textual priors. Instead of conditioning only on a partial occupancy map, the predictor could also take natural language prompts (e.g., "kitchen sink with pipes" or "bookshelf with partitions"). As proof of concept, we extended our CNN predictor with CLIP-based prompt embeddings [163], injected via FiLM modules [164] at each encoder stage. This text conditioning enabled the CNN to bias its predictions toward the described domain and, when integrated with the planner, the model performed competitively with the baseline CNN, achieving average task completion times of 216 seconds (pipes) and 157 seconds (shelves) in simulation. These results highlight the potential of textual priors.

In conclusion, we introduced a theoretically complete and empirically efficient framework for manipulating using contacts. The framework tightly couples three components—torque-based contact detection and localization, workspace occupancy estimation, and planning—while carefully reasoning about noise and uncertainty in both contact localization and occupancy prediction. Experiments in both simulation and in the real world show that our framework reliably completes tasks such as valve manipulation and object retrieval, achieving up to a 2× reduction in task completion time compared to baselines.

IV

Conclusion

7 Conclusion

7.1 Future Work

There are numerous avenues for further research building on the contact-driven frameworks developed in this thesis. The range of capabilities that can be enabled through contact-driven skills is broad, and we believe that current approaches are still in an early stage of development. In this section, we outline several promising directions for future work, including ideas that were briefly investigated in this thesis.

7.1.1 Semantic and task-conditioned occupancy predictions

The goal of the occupancy predictor developed in Chapter 6 was to use the history of proprioceptive signals to construct a partial occupancy map of the workspace, which is then extrapolated to predict the occupancy of the entire workspace using a learned model trained on a distribution of plausible environments for a given domain. For the valve manipulation task, the predictor was trained on a distribution of kitchen sink workspaces consisting of pipes arranged in varying configurations. For the object retrieval task, the training data consisted of typical shelf geometries. A single predictor was trained across both domains, with a conditional flag provided as input to indicate the domain being solved.

A promising direction for future work is to remove this explicit domain flag and instead condition the occupancy predictor on textual priors. Rather than conditioning solely on the partial occupancy map, the predictor could additionally take natural language prompts such as "kitchen sink with pipes" or "bookshelf with partitions." As a proof of concept, we extended our CNN-based predictor with CLIP-derived text embeddings [163], which were injected via FiLM modulation layers [164] at each encoder stage. We used a set of 50 distinct prompts, with 25 prompts per domain. This form of text conditioning enabled the network to bias its predictions toward environment structures consistent with the provided semantic description.

When integrated with the planner, the text-conditioned predictor performed competitively with the baseline CNN, achieving average task completion times of 216 seconds for the valve manipulation

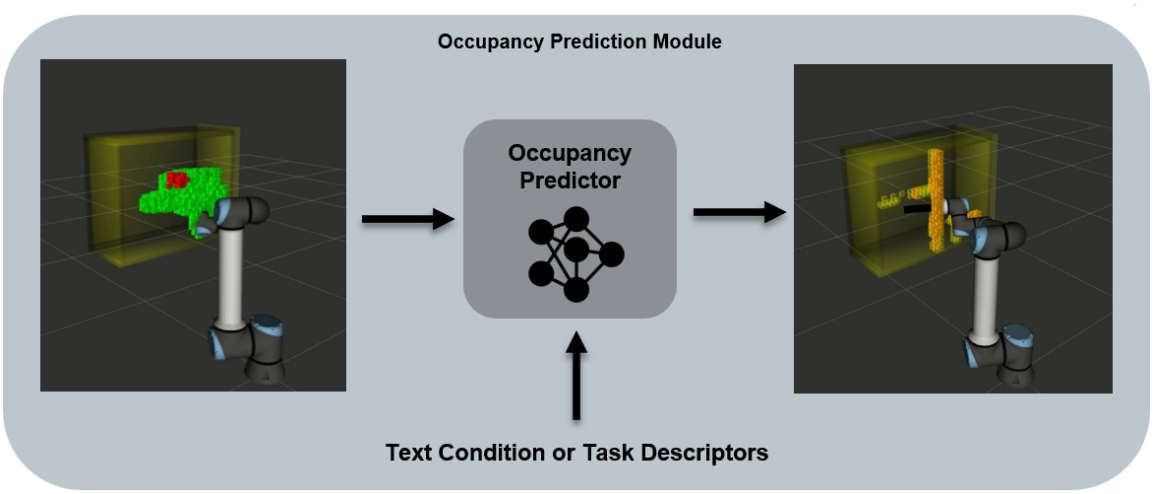

Figure 7.1: Task/text-conditioned occupancy predictor. The predictor takes a partial occupancy map of the workspace, constructed from the history of proprioceptive observations, together with textual priors (e.g., "There is a pipe in the middle of the shelf") or other task descriptors, and outputs a complete workspace occupancy map.

task and 157 seconds for the shelf retrieval task in simulation. These results suggest that textual priors can serve as an effective mechanism for encoding domain knowledge and guiding occupancy prediction in contact-driven manipulation.

While the text conditioning used in this work was deliberately simple and intended primarily as a proof of concept, it highlights several promising avenues for future research. More expressive text-conditioned models could be developed that reason more explicitly about spatial semantics conveyed through language. For example, given a prompt such as "there is a pipe in the middle of the shelf," the predictor should assign high occupancy probability to cells near the center of the workspace, even in the absence of direct contact evidence. Such models would allow semantic information to shape spatial predictions in a more structured and interpretable manner, further bridging high-level task descriptions and low-level geometric reasoning.

The preliminary results on text-conditioned occupancy prediction suggest a broader direction of incorporating semantic and task-level priors into contact-driven world modeling. Rather than treating occupancy prediction as a purely geometric inference problem, future work can frame it as a semantic world modeling problem, where predictions are shaped jointly by partial observations, contact history, and high-level task context.

One promising extension is to condition occupancy predictors not only on free-space observations and contact events, but also on task descriptions, object categories, or expected interaction patterns. For example, a task description such as "turn the shutoff valve" implicitly conveys structural information about the workspace, including the likely presence of pipes, brackets, and constrained free space near the target. Conditioning world models on such task semantics could bias predictions toward physically and functionally plausible environments, even in regions that have not yet been explored through contact.

More generally, let $\mathcal{D}$ denote a domain descriptor that may include natural language prompts,

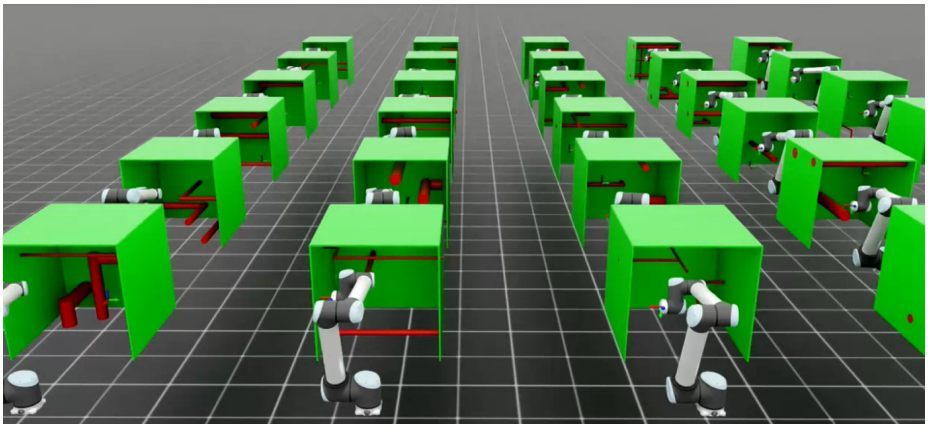

Figure 7.2: Visualization of the RL environments for the blind manipulation task in IsaacSim [165].

symbolic task specifications, or learned semantic embeddings. An occupancy predictor can then be formulated as $p(W \mid \mathcal{O}, \mathcal{C}, \mathcal{D})$, where $\mathcal{O}$ denotes partial free-space observations and $\mathcal{C}$ denotes the history of contact events. This formulation naturally subsumes the text-conditioned CNN model as a special case and provides a flexible interface for integrating multimodal priors into the belief update process.

7.1.2 Reinforcement learning for manipulation in the blind

In Chapter 6, we developed a contact-driven framework that enables a robot to navigate from a start configuration to a goal configuration in partially known or unknown environments using contact interactions with the workspace. While this model-based framework achieved strong empirical performance, it relies on several explicit design choices whose optimality is not guaranteed. These include representing workspace uncertainty using an occupancy grid, hand-designing cost functions that combine multiple geometric and heuristic components, and specifying how contact information should be incorporated into the planning process.

Such choices may introduce inductive biases that limit performance or generalization. Although contact observations and task-specific parameters such as robot and object geometry are available, the intermediate representations used to reason about the workspace are explicitly constructed rather than learned. This raises a fundamental question: *can a robot learn a direct mapping from contact observations to actions, without relying on hand-designed intermediate representations?*

One potential advantage of a learned approach is the ability to internally construct latent representations that are tailored to the task objective, rather than being constrained by pre-specified abstractions such as occupancy grids. In this setting, the policy can implicitly learn which aspects of the contact history are relevant for navigation and how they should influence future actions.

This motivates the use of reinforcement learning to learn navigation policies directly from interaction. Supervised learning is not well-suited for this problem due to the absence of expert demonstrations. Moreover, learning from trajectories generated by a hand-designed planner risks inheriting its suboptimalities. Reinforcement learning, in contrast, enables policies to emerge through

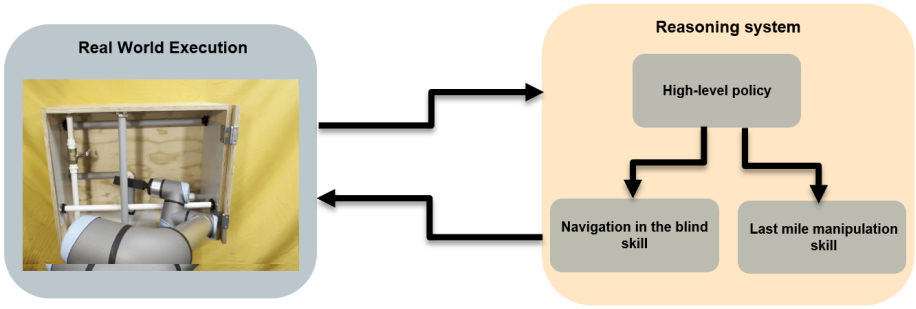

Figure 7.3: A hierarchical closed-loop reasoning and execution framework. Based on the history of observations made, a higher-level orchestrator policy picks between a navigation in the blind skill and a last-mile manipulation skill for execution.

episodic interaction with the environment, optimizing task performance directly without requiring ground-truth actions.

As a preliminary step in this direction, we developed an initial reinforcement learning framework for navigation in the blind using Soft Actor-Critic (SAC). The policy is conditioned on the history of robot configurations and observed contact events, and outputs joint-space action increments. To facilitate learning in this challenging contact-rich domain, we employed a curriculum learning strategy that gradually increases task difficulty. Training begins in empty workspaces and progressively introduces structured obstacles, starting with a single pipe in a kitchen sink cabinet and gradually increasing the number and complexity of pipes.

Using this curriculum, we achieve success rates of approximately 50% on the full navigation task in complex, cluttered environments. While these results are preliminary, they indicate the feasibility of learning contact-driven navigation policies without explicit workspace representations. There remain many open directions for improving performance, which represent promising avenues for future research.

7.1.3 A hierarchical framework for object manipulation in the blind

The frameworks developed in the preceding chapters address two complementary but isolated sources of uncertainty in manipulation. In Chapters 3 and 4, we focused on reducing uncertainty over the *pose of a target object* while assuming a fixed and known workspace. In Chapter 6, we addressed the dual problem of manipulating in an *unknown or partially known workspace* while assuming the target object pose to be fixed or known.

In real-world manipulation, however, these assumptions rarely hold independently. Robots are often required to operate in environments that are both cluttered and poorly perceived, while simultaneously reasoning about uncertainty in the pose or configuration of the object they seek to manipulate. Human behavior in such settings naturally couples these two reasoning processes. Consider the example of reaching a shutoff valve beneath a kitchen sink. A human initially navigates through an uncertain and cluttered workspace toward an approximate estimate of the valve location. As contacts are encountered, the human reasons about likely obstacle structures such as pipes and

brackets and adapts their path accordingly. Once the hand reaches the vicinity of the valve, tactile feedback is then used to infer the valve's precise configuration and enable the desired manipulation.

This observation motivates a unified framework that reasons jointly about workspace uncertainty and object pose uncertainty. We view the two classes of methods developed in this thesis as *skills* that operate at different levels of abstraction and can be composed hierarchically. The contact-driven workspace reasoning framework developed in Chapter 6 can be interpreted as a navigation skill that enables the robot to safely and efficiently reach the vicinity of a target region under severe environmental uncertainty. The contact-driven object localization and manipulation frameworks developed in Chapters 3 and 4 can be viewed as object-centric manipulation skills that refine pose uncertainty and execute high-precision actions once the robot is sufficiently close to the target.

A natural direction for future work is to integrate these skills within a hierarchical reinforcement learning or hierarchical decision-making framework. At the high level, a manager policy would reason over coarse task progress and select between navigation-oriented skills that reduce workspace uncertainty and manipulation-oriented skills that reduce object pose uncertainty. At the low level, each skill would invoke a model-based or a learned policy for both point-to-point navigation in the blind and for contact localization for object-centric manipulation.

Formally, this setting can be modeled as a hierarchical decision-making problem with two low-level skills and a high-level coordination policy. Let the robot configuration space be denoted by $\mathcal{Q}$, the workspace state by W, and the pose of the target object by $h \in SE(3)$.

We define two low-level policies (skills):

1. **Blind Navigation Skill** $\pi_{\text{nav}}(q_{\text{goal}})$: A contact-driven policy that attempts to move the robot from its current configuration $q \in \mathcal{Q}$ to a specified goal configuration $q_{\text{goal}} \in \mathcal{Q}$ in the presence of unknown obstacles. This policy reasons primarily over workspace uncertainty b_W and uses contact feedback to infer free and occupied regions of the environment, as developed in Chapter 6. Its termination conditions include reaching q_{goal} or observing a contact.

2. **Last Mile Manipulation Skill** π_{obj}: A contact-driven policy that actively reduces uncertainty over the object pose belief b_h and executes a task-completion maneuver once uncertainty falls below a task-dependent threshold. This policy is similar to the POMDP-based localization and manipulation frameworks developed in Chapters 3 and 4. Examples of tasks that could be of interest include valve manipulation, peg insertion, and switch manipulation.

At the high level, we introduce a manager policy Π_{HL} that operates over an abstract state defined by the joint belief b and coarse task progress indicators. The role of Π_{HL} is to select which low-level skill to execute and to parameterize the selected skill.

Execution should proceed as follows. Initially, given a coarse prior over the object pose b_h and a highly uncertain workspace belief b_W, the high-level policy selects a navigation action $(\text{nav}, q_{\text{goal}})$, where q_{goal} is chosen to lie near the expected object location under b_h. While $\pi_{\text{nav}}(q_{\text{goal}})$ is executing, the robot accumulates contact observations that update b_W. If the navigation policy terminates unsuccessfully due to contacts with the environment, the high-level policy updates its belief and selects an alternative goal configuration q'_{goal}, thereby re-invoking $\pi_{\text{nav}}(q'_{\text{goal}})$.

Once the robot reaches a configuration sufficiently close to the expected object location and the uncertainty in b_W around the target region is reduced, the high-level policy transitions to the object-centric skill π_{obj}. This skill actively reasons over b_h using contact observations with the object of interest to localize its pose and complete the manipulation task.

A key challenge in this hierarchical setting is that contact observations are inherently ambiguous. A contact event z_{contact} may arise either from interaction with a workspace obstacle or from interaction with the target object. Consequently, belief updates must reason jointly about both hypotheses. During navigation, contacts are more likely attributed to workspace obstacles and primarily update b_W. During object localization, contacts near the expected object region are more likely attributed to the target object and update b_h. The high-level policy can further disambiguate these cases by conditioning belief updates on the active skill and its intended objective.

This hierarchical formulation provides a principled way to couple workspace exploration and object-centric manipulation while retaining the benefits of specialized reasoning at each level. By decomposing the problem into reusable low-level skills and a belief-aware high-level coordinator, the framework enables scalable decision-making under joint uncertainty. Given the lack of ground-truth supervision for such tasks, a hierarchical reinforcement learning setup that learns from repeated interaction and execution experience is a particularly well-suited direction for future work.

7.1.4 Non-static obstacles

Throughout this thesis, we focused on manipulation scenarios in which the objects involved in contact interactions were assumed to be static. In the case of manipulation under object pose uncertainty, the pose of the object of interest was assumed to remain fixed during interaction. Similarly, in the case of manipulation in uncertain environments, workspace obstacles were assumed to be static. While these assumptions are valid for many real-world tasks and are appropriate for the domains studied in the preceding chapters, there exist numerous practical scenarios in which the objects being interacted with are not stationary.

Relaxing the static-object assumption introduces the need to explicitly reason about the dynamics of contact interactions and the resulting motion of objects in the environment. In such settings, contacts no longer provide only geometric constraints but also induce state transitions in the environment that must be estimated and predicted. This substantially increases the complexity of belief estimation and planning, as the robot must now reason jointly about contact location, contact force, and object motion.

This challenge is particularly pronounced for precision tasks such as insertion, where allowable tolerances are small. If the object being manipulated, such as a port, moves as a result of contact, the robot must accurately infer how the object has shifted in response to the interaction. Even small, unmodeled motions can cause task failure, making robust insertion under object mobility a significantly harder problem than the static case.

Non-static obstacles pose challenges in the context of navigation in the blind as well. In this setting, contacts can occur anywhere along the robot arm, and in our current framework, interaction forces are inferred indirectly through joint current measurements. As a result, the forces applied

to the environment are typically larger and less controlled than in object pose uncertainty tasks, where a wrist-mounted force-torque sensor enables more delicate interaction. When obstacles are mobile, such forceful contacts can easily induce unintended object motion, further complicating belief estimation and planning. We hypothesize that reliably manipulating in unknown environments with mobile obstacles will require richer and more sensitive tactile sensing modalities. Human manipulation in cluttered environments relies heavily on sensitive tactile feedback to regulate contact forces and infer object motion. Analogously, equipping robots with tactile skins or high-resolution contact sensors [135, 158] along the arm could enable more precise regulation of interaction forces and provide richer observations for estimating how objects respond to contact. Such sensing would allow robots to better model contact dynamics, distinguish between different interaction regimes, and update beliefs over object motion more accurately.

7.1.5 Accounting for other sources of uncertainty

In Part I of this thesis, we focused on reasoning under uncertainty in the pose of the object of interest. However, pose uncertainty represents only one of several sources of uncertainty that arise in contact-driven manipulation. In many realistic settings, the robot may also be uncertain about other object properties, such as geometry, dimensions, articulation parameters, or even object identity. Accounting for these additional uncertainties can lead to richer behaviors and more robust manipulation strategies.

One particularly relevant extension is joint uncertainty over object pose and object model. In practice, a robot may possess only an approximate or incomplete model of the object it seeks to manipulate. For example, the robot may know that the object of interest is a port with a roughly cylindrical structure, but lack precise information about its dimensions or tolerances. In such cases, the robot must reason simultaneously about where the object is located and what its exact geometry is. This requires maintaining and updating a belief not only over pose variables, but also over latent object parameters that define the object model.

Formally, this can be viewed as maintaining a belief over an augmented state space that includes both pose and model parameters. Contacts that are inconsistent with the assumed geometry can reduce the probability of certain parameter values, allowing the robot to refine its internal object model over time. Extracting informative object features from interaction is a key challenge in this setting. Rather than reasoning over full geometric models, the robot may need to infer coarse, task-relevant features such as opening width, depth, or symmetry. These features can then be matched against a library of known object classes or parameterized templates. For instance, repeated contacts during an insertion attempt could reveal whether a port is narrower or wider than expected, enabling the robot to adapt its strategy accordingly.

These ideas naturally extend to object identification tasks. In cluttered environments such as shelves or cabinets containing multiple objects, the robot may be uncertain about which object is the target. In such cases, contact-driven interaction can be used not only to localize objects, but also to disambiguate object identity by comparing observed interaction patterns against expected behaviors for different object hypotheses.

7.2 Limitations

While the frameworks presented in this thesis demonstrate strong robustness and generality, they also have several important limitations.

- First, the contact sensing modalities utilized are relatively primitive. Our framework relies primarily on binary contact detection, robot configurations at which contacts occur, and joint-current–based estimates of contact location. Although these signals are robust and widely available on modern manipulators, they are inherently sparse and low-dimensional. As a result, the robot often requires many interactions to accurately infer object pose or workspace structure, leading to increased execution time. Richer tactile sensing modalities, such as high-resolution force–torque sensing or whole-arm tactile skins, could provide denser information and significantly reduce the number of exploratory contacts required, potentially enabling more human-like manipulation speeds.

- Second, throughout this thesis we assume that objects being interacted with are static. While this assumption holds in many practical settings, there are numerous real-world scenarios where objects may move as a result of contact. Reasoning about object dynamics under contact, particularly in cluttered environments, would require more accurate modeling of contact forces and object motion, and remains a challenging extension of the current framework.

- Third, although the underlying problems are naturally formulated as POMDPs, solving POMDPs exactly or even approximately remains computationally challenging as uncertainty scales. While we introduced several structural approximations, hierarchical representations, and heuristic strategies to make planning tractable, there may exist regimes where explicit POMDP reasoning becomes impractical. In such cases, alternative formulations that combine model-based reasoning with learned policies or implicit belief representations may be more appropriate.

- Throughout the thesis, we assume that robot transitions are deterministic and noise-free. While this is a reasonable approximation for high-precision industrial manipulators such as the UR10e used in our experiments, it may not hold for other robotic platforms operating under actuation uncertainty, compliance, or external disturbances. In such settings, transition noise must be explicitly incorporated into the system dynamics model. Similarly, in the earlier portions of this thesis we deliberately relied on a simple yet robust binary contact signal, which allowed us to reasonably assume near-perfect observations. However, richer sensing modalities, such as continuous force or tactile feedback, are inherently noisier and would require probabilistic observation models. Introducing stochastic transitions and observations increases branching in the belief space, substantially enlarging the search space and making planning more computationally demanding. Extending the proposed methods to efficiently handle fully stochastic transition and observation models remains an important direction for future work.

- In addition, the uncertainty considered in this thesis is often structured. For example, we assume that the object model is known while its pose is uncertain. In many domestic and

unstructured environments, however, uncertainty may exist simultaneously over object pose, geometry, articulation, and identity. Reasoning over such unstructured and high-dimensional uncertainty would require richer belief representations and more expressive observation models than those considered here.

- Many components of the proposed systems rely on hand-designed representations and heuristics, such as occupancy grids, cost functions, and belief update rules. While these choices provide interpretability and enable principled reasoning, they also impose inductive biases that may be suboptimal in certain domains. Learning-based approaches that directly map contact histories to actions or latent representations could alleviate these constraints and improve performance and generalization, albeit at the cost of reduced interpretability and increased data requirements. In addition, online planning with model-based methods incurs nontrivial computational overhead at execution time. A promising direction is the development of hybrid systems in which learned models handle the majority of common interactions efficiently, while model-based planners are invoked selectively to handle complex or safety-critical cases in the tail of the task distribution. Such hybrid systems could further incorporate multiple learned skills, enabling flexible composition of behaviors within the overall planning framework.

Despite these limitations, we believe the presented work establishes a strong foundation for contact-driven manipulation and highlights several promising directions for future research.

7.3 Broader applicability of the developed solutions

Although this thesis focuses on contact-driven manipulation in the blind, several of the algorithmic contributions were deliberately developed to be domain-agnostic and broadly applicable beyond the specific manipulation settings considered here. In particular, the core planning contributions in the first part of the thesis were designed as general-purpose methods for solving POMDPs more efficiently.

Experience-Based POMDP Planning (Chapter 3): The idea underlying Experience-based RTDP-Bel (E-RTDP-Bel) is fundamentally general: similar planning problems are likely to have similar solutions, and previously computed policies can be reused to accelerate future planning queries. This principle is not tied to contact-based manipulation. It applies to any setting where a sequence of related POMDPs must be solved, such as lifelong robotic operation [166, 166] or the offline construction of a solution database [16], as explored in our work. In fact, this idea was originally explored in deterministic planning settings [89], and in this thesis, we extended it to the stochastic planning regime.

While the underlying idea is general, it is important to discuss the specific solver E-RTDP-Bel as well. E-RTDP-Bel does not exploit manipulation-specific structure. It constructs an experience heuristic from a base admissible heuristic and the solutions of similar problems, which we refer to as experience. This experience augmented heuristic is then used to bias search toward promising regions of the belief space. Consequently, in any domain where heuristic search solvers are suitable

and a sequence of related or structurally similar POMDP problems arises, the planner can be directly deployed, provided a suitable base heuristic can be defined. The method, therefore, transfers naturally to other robotic and non-robotic domains characterized by recurring problem structure.

Further, the approach is not confined to RTDP-Bel [71]. The idea of experience-augmented heuristics can, in principle, be incorporated into other heuristic search-based POMDP solvers [72, 125]. Designing and analyzing such experience-based variants, both theoretically and empirically, is a promising direction for future work.

Lazy Heuristic Search for POMDPs (Chapter 5): The second major algorithmic contribution, lazy heuristic search for POMDPs, was explicitly developed as a general solution to a common computational bottleneck: expensive belief transition evaluation. In contact-rich manipulation, belief transitions require expensive collision checks. In indoor navigation, they require raycasting. In outdoor navigation, they may require physics-based simulation. In other domains, they may require high-fidelity dynamics models, complex sensor models, or large-scale probabilistic updates. The lazy search principle applies uniformly across all such cases.

The developed solvers, Lazy RTDP-Bel and Lazy LAO*, are not specific to manipulation. They can be applied to any domain where heuristic search solvers are effective, and belief transitions are computationally expensive, so long as a fast-to-query and reasonably informative Q-estimator can be constructed. When these conditions hold, laziness provides computational savings by deferring expensive evaluations until they are necessary. We also provide domain-independent techniques for constructing useful estimators, including subsampling-based estimators and MDP-inspired estimators, which can be adapted to many other problems (Chapter 8).

The idea of deferring belief transition computations using action value estimates also extends naturally to other heuristic search solvers and even to fully observable MDP settings (Section 5.3.4). Similar to experience-based planning, lazy search was also originally developed in deterministic contexts [121, 122] and is here extended to stochastic belief-space planning.

Hierarchical Uncertainty Representation (Chapter 4): While the developed hierarchical belief representations were motivated by large-scale pose uncertainty in contact-rich manipulation, the underlying principle is more general: uncertainty can be represented at multiple levels of abstraction, with reasoning beginning in coarse representations and progressively refining as needed. This idea is applicable in domains such as multi-resolution mapping [167, 168], large-scale localization [169], target tracking [170], and exploration under uncertainty [171]. Whenever belief representations are high-dimensional or combinatorial, structured abstractions can reduce computational burden while preserving task-relevant information.

Taken together, the contributions of this thesis extend beyond the specific application of manipulation in the blind. Although the empirical evaluations focused on contact-driven manipulation, the underlying algorithms and ideas are not tied to any specific sensing modality or robot embodiment. In this sense, while the motivating theme of the thesis is contact-driven manipulation, the planning methodologies developed here form a broader contribution to scalable decision-making under uncertainty.

7.4 Conclusion

This thesis investigated how robots can leverage contact feedback to perform reliable manipulation in settings where visual perception is insufficient, unreliable, or unavailable. Across a spectrum of environments—from semi-structured industrial workcells to unstructured domestic settings and fully unknown cluttered workspaces—we demonstrated that contact-driven reasoning provides a powerful mechanism for handling uncertainty in manipulation.

A central theme throughout this work is the explicit treatment of manipulation as decision-making under uncertainty, where contacts are not merely disturbances to be avoided, but informative observations that can be actively exploited. By formulating these problems within principled planning frameworks and carefully designing representations that respect both the structure of uncertainty and the limitations of real robotic systems, this thesis shows that high-precision and robust manipulation is possible even under severe perceptual constraints.

- In Chapter 3, we addressed high-precision insertion tasks in semi-structured industrial environments, where object pose uncertainty is small but task tolerances are extremely tight. By formulating contact-based active localization as a POMDP and exploiting the finite and known nature of the uncertainty space, we introduced a preprocessing-based planning framework that enables fast and reliable online execution.

- Chapter 4 extended this line of work to unstructured domestic environments, where both the scale and diversity of uncertainty make offline preprocessing infeasible. To address this challenge, we introduced a hierarchical representation of uncertainty that enables reasoning at multiple levels of abstraction. By first operating in a coarse volumetric space and later transitioning to a finer particle-based representation, the framework dramatically reduces computational complexity while preserving decision quality. Combined with a closed-loop, anytime planning framework, this approach enabled online solution synthesis for large pose uncertainties that would otherwise be intractable.

- A common computational challenge in the frameworks introduced in Chapters 3 and 4 is that belief transitions in contact-rich manipulation as well other robotics domains is often expensive to compute. Chapter 5 addressed this challenge directly by introducing lazy heuristic search methods for POMDP planning with expensive belief transitions. We proposed Lazy RTDP-Bel and Lazy LAO*, which defer belief transition computation until it is necessary, using fast-to-query Q-value estimators to guide search. These methods significantly reduce planning time while preserving solution quality and optimality guarantees.

- Finally, Chapter 6 tackled a fundamentally different but complementary problem: manipulation in unknown or partially known workspaces, where uncertainty lies not in the pose of a target object but in the structure of the environment itself. Here, we presented a complete contact-driven framework that integrates torque-based contact detection and localization, learned occupancy prediction, and robust planning. By combining sparse but reliable binary contact detections with learned structural priors and theoretically grounded planning methods,

the framework achieves both empirical efficiency and formal completeness guarantees. Real-robot experiments on valve manipulation and object retrieval tasks underscored the value of explicitly reasoning about workspace uncertainty.

Beyond the specific domain of contact-driven manipulation, this thesis also contributes more broadly to planning under partial observability. We developed theoretically grounded POMDP algorithms that extend beyond the manipulation settings considered here. In particular, we introduced Experience-based RTDP-Bel, which leverages solutions to related problems to accelerate planning, and Lazy RTDP-Bel, which reduces computational overhead by postponing expensive belief transition evaluations. These contributions address fundamental scalability challenges in POMDP planning and are applicable to a wide range of robotics problems.

Looking forward, this work opens several promising avenues for future research. Integrating richer sensing modalities, reasoning about dynamic objects and contacts, extending uncertainty modeling to object identity and geometry, and developing hybrid learning-and-planning systems all represent natural next steps. More broadly, the ideas presented in this thesis suggest that contact-driven skills can serve as foundational building blocks for robust robotic manipulation in real-world settings, enabling robots to operate effectively in environments that remain challenging for purely vision-based systems.

In conclusion, this thesis establishes contact as a powerful sensing and reasoning modality for robotic manipulation and provides a set of algorithms, representations, and empirical results that advance regime of manipulation under uncertainty. We hope that these contributions help pave the way toward more capable, adaptable, and resilient robotic systems operating beyond controlled laboratory environments.

V

Appendix

8 Q-Estimators for Lazy Heuristic Search

Here, we provide details about the Q-estimators, $\hat{Q}_{init}$, used for the problem domains we evaluated our framework on. We also discuss and evaluate a probabilistically conservative estimator for the manipulation using contacts task to highlight the scope of estimators that can be developed.

8.0.1 Manipulation for pose estimation using contacts

The goal is to localize a target object T, which in this case is a charger plug, and complete a plug insertion task. However, the pose of the plug is not directly observable, instead, the robot manipulator needs to use binary contact observations to infer the pose. Let the particle set $H = \{h_1, h_2, \dots\}$ represent the distribution of hypothesis poses for the target object T. The action space includes unit robot movements and compliant controllers that track object surfaces upon contact. The observation space consists of the robot state q (from encoder readings) and a binary collision signal (from an F/T sensor). The belief state is given by $b = \{q, H\}$. The objective is to find a policy that fully localizes the target object (belief state with $|H| = 1$) while minimizing the robot's expected distance travelled.

Given a target object pose h_i, the expected observation—i.e., the sequence of contact signals and robot states—must be computed by forward simulating the action with the object model at pose h_i. This involves complex mesh-to-mesh collision checks to detect exact contact points and compute the trajectory tracked by the compliant controller. To compute the successor beliefs for a given belief state, the action must be forward simulated for all hypothesis poses $h_i \in H$. This makes evaluating an action and computing successor beliefs (and their probabilities) computationally expensive, especially when the belief state contains thousands of hypothesis poses.

Given a belief state $b = \{r, H\}$, an action a, and an observation z, the corresponding successor belief $b_a^z = \{r', H_a^z\}$ consists of all hypothesis poses under which z would have been observed when executing a from r. Since the particles in the hypothesis set are unweighted, the probability of observing z is given by $P(z|b, a) = \frac{|H_a^z|}{|H|}$. Since this is an active localization problem, we employ a scaled version of the entropy of a belief state, measured by the size of the hypothesis set $|H|$, as our heuristic function. Specifically, we define the heuristic as $\text{heur}(b_a^z) = \alpha|H_a^z|$, where α is a scaling factor. This formulation reflects the intuition that a larger number of particles in the current belief

state corresponds to increased localization effort. Consequently, we can rewrite Q_{init} as follows:

$$\begin{aligned} & Q_{init}(b,a) = \mathcal{C}(b,a) + \sum_{z \in \mathcal{Z}} P(z|b,a)\,\text{heur}(b_a^z) \\ \implies & Q_{init}(b,a) = \mathcal{C}(b,a) + \sum_{z \in \mathcal{Z}} \frac{|H_a^z|}{|H|}(\alpha |H_a^z|) \\ \implies & Q_{init}(b,a) = \mathcal{C}(b,a) + \alpha \sum_{z \in \mathcal{Z}} \frac{|H_a^z|^2}{|H|} \end{aligned} \tag{8.1}$$

Hence, if we can compute $|H_a^z|$, then we can compute $Q_{init}(b,a)$. However, computing $|H_a^z|$ would involve rolling out the action on all particles in H.

Now, the objective is to subsample a belief state $\hat{b} = \{r, \hat{H}\}$ from b, evaluate the action on the subsampled belief state, compute $\hat{H}_a^z$, and utilize it to estimate H_a^z. For notational simplicity, let n represent the number of particles in H. The subset $\hat{H}$ is created by sampling k particles from H, where $k \ll n$. Hence, evaluating the action on $\hat{b}$ and computing $\hat{H}_a^z$ is significantly faster.

If we assume that the observations obtained when executing a on different particles are independent (a simplifying assumption commonly employed), we can view the problem as follows: each particle in H contains a hidden variable representing its assignment to one of $|\mathcal{Z}|$ groups, where the assignment represents the observation obtained when executing a on the particle. The goal is to predict the distribution, i.e., the number of particles in each group, when the assignments of only a subset of the particles are visible. Based on this perspective, $\frac{n}{k}\hat{H}_a^z$ serves as an unbiased estimator of H_a^z. Since the particles are randomly chosen to be part of $\hat{H}$, the probability of a particle being in $\hat{H}$ is given by $\frac{k}{n}$, and since the observations of the particles are independent, we can write:

$$\begin{aligned} \mathbb{E}_{\hat{H} \sim H}\left(|\hat{H}_a^z|\right) = \frac{k}{n}|H_a^z| \implies \mathbb{E}_{\hat{H} \sim H}\left(\frac{n}{k}|\hat{H}_a^z|\right) \\ = |H_a^z| \end{aligned} \tag{8.2}$$

This implies that we can write the unbiased subsampled estimator for the manipulation problem as:

$$\hat{Q}_{init}(b,a) = \mathcal{C}(b,a) + \alpha \left(\frac{n}{k}\right)^2 \sum_{z \in \mathcal{Z}} \frac{|\hat{H}_a^z|^2}{|H|} \tag{8.3}$$

As we can see, this is simply a minor manipulation of $Q_{init}(\hat{b},a)$, which can be written as:

$$Q_{init}(\hat{b},a) = \mathcal{C}(b,a) + \alpha \sum_{z \in \mathcal{Z}} \frac{|\hat{H}_a^z|^2}{|H|} \tag{8.4}$$

The estimator $\hat{Q}_{init}(b,a)$ essentially contains a correction term in the form of the squared sampling ratio $\left(\frac{n}{k}\right)^2$, to make the estimate unbiased and effective for our problem domain.

Probabilistically Conservative Estimator

While the subsampled estimator defined in Eq. 8.3 is an effective and unbiased approximation, it does not guarantee conservativeness, i.e., $\hat{Q}_{init}(b,a)$ is not necessarily less than or equal to $Q_{init}(b,a)$. Since the heuristic in this problem is based on entropy which is not an admissible heuristic, there might not be a need for a conservative estimator. Nonetheless, we experimented with a modified version of the subsampled estimator that makes it probabilistically conservative, i.e., with 95% confidence $\hat{Q}_{init}(b,a) \leq Q_{init}(b,a)$.

To achieve this, we leverage Hoeffding's inequality. Given independent Bernoulli random variables X_i, let $S_n = \sum X_i$. Hoeffding's inequality states:

$$P(S_n - \mathbb{E}[S_n] \geq \epsilon) \leq \exp\left(\frac{-2\epsilon^2}{n}\right). \tag{8.5}$$

If we assume the membership of the particles to each of the different observations to be independent (as we have previously), we can write,

$$P\left(|\hat{H}_a^z| - |H_a^z|\frac{k}{n} \geq \epsilon\right) \leq \exp\left(\frac{-2\epsilon^2}{k}\right). \tag{8.6}$$

Setting $\epsilon = 1.22\sqrt{k}$, we obtain:

$$P\left(\left(|\hat{H}_a^z| - 1.22\sqrt{k}\right)\frac{n}{k} \geq |H_a^z|\right) \leq 0.05. \tag{8.7}$$

This result implies that with 95% confidence, the term $(|\hat{H}_a^z| - 1.22\sqrt{k})\frac{n}{k}$ provides a conservative estimate of H_a^z. Consequently, we define the probabilistically conservative subsampling estimator as:

$$\hat{Q}_{init}(b,a) = \mathcal{C}(b,a) + \alpha \sum_{z \in \mathcal{Z}} \frac{\left((|\hat{H}_a^z| - 1.22\sqrt{k})\frac{n}{k}\right)^2}{|H|}. \tag{8.8}$$

Thus, Eq. 8.8 defines an estimator that is conservative with 95% confidence. We believe there is significant potential to explore similar and innovative estimators for common robotics problems. This example was simply to highlight the scope. Additionally, we also evaluated the conservative estimator on the manipulation using contacts task, the results for which have been included in Table 8.1. We represent this probabilistically conservative estimator as $\hat{Q}_{init}^{pce}$, while the vanilla subsampling estimator is continued to be represented with $\hat{Q}_{init}$.

8.0.2 Navigation

For the information-gathering version of the navigation problem, we defined the heuristic, identical to the manipulation problem to be the scaled version of the number of particles in the current belief. Hence, the subsampled estimator $\hat{Q}_{init}$ was defined identically as in Eqn. 8.3 (which includes the additional bias correction term).

Planner	**Success Rate**	**Planning Time**	**Cost**
RTDP-Bel	0.87	307.49	0.69
Lazy RTDP-Bel ($\hat{Q}_{init}$)	1.0	183.18	0.71
Lazy RTDP-Bel ($\hat{Q}^{pce}_{init}$)	1.0	221.20	0.71
LAO*	0.93	253.67	0.69
Lazy LAO* ($\hat{Q}_{init}$)	1.0	136.73	0.70
Lazy LAO* ($\hat{Q}^{pce}_{init}$)	1.0	175.14	0.72

Table 8.1: Performance comparison on the manipulation for pose estimation problem.

For the goal-directed version of the problem, the subsampled estimator was defined as outlined in Section 4 of the paper (without any additional bias correction terms).

$$\hat{Q}_{init}(b, a) = Q_{init}(\hat{b}, a) \tag{8.9}$$

The Q_{MDP} estimator, $\hat{Q}^{MDP}_{init}$ was also utilized as defined in Section 4 of the paper,

$$\hat{Q}^{MDP}_{init}(b, a) = \sum b(s) \left(c(s, a) + \sum_{s'} T(s, a, s') \text{heur}(s') \right) \tag{8.10}$$

8.0.3 Subsampling Estimator

The subsampling estimator is effective in its standard form when the heuristic satisfies

$$\text{heur}(b) = \mathbb{E}_{s \sim b} \text{heur}(s).$$

Many heuristics used in robotics planning, particularly in goal-directed tasks (e.g., an autonomous ground vehicle navigating to a goal), adhere to this property. While effective, the standard subsampling estimator is not truly unbiased due to dependencies between variables in the estimation process. To obtain a truly unbiased estimate, we must decompose the initial Q-value expression into independent subexpressions and estimate each component separately.

The initial Q-value is defined as

$$Q_{\text{init}}(b, a) = \mathcal{C}(b, a) + \sum_{z \in Z} P(z|b, a) \cdot \text{heur}(b^z_a), \tag{8.11}$$

where:

- $\mathcal{C}(b, a)$ represents the cost of executing action a in belief b.
- $P(z|b, a)$ is the probability of receiving observation z after taking action a.
- $\text{heur}(b^z_a)$ is the heuristic value of the updated belief after observation z.

Sources of Bias

Directly estimating $Q_{\text{init}}(b, a)$ using the subsampling approach introduces bias due to the non-linear dependencies within the summation. The observation probability $P(z|b, a)$ is given by:

$$P(z|b, a) = \sum_s b(s) \sum_{s'} T(s, a, s')O(s', a, z), \tag{8.12}$$

which depends on the original belief distribution $b(s)$. Similarly, the heuristic term $\text{heur}(b_a^z)$ expands as:

$$\text{heur}(b_a^z) = \sum_s b_a^z(s)\text{heur}(s), \tag{8.13}$$

where

$$b_a^z(s) = \frac{\sum_{s'} T(s', a, s)O(s, a, z)b(s')}{P(z|b, a)}. \tag{8.14}$$

Both the numerator and denominator of $b_a^z(s)$ are functions of $b(s)$, leading to statistical dependencies that introduce bias when estimated jointly.

Unbiased Estimation

To eliminate this bias, we construct independent estimates for three key components:

1. The observation probability $P(z|b, a)$.
2. The numerator of $\text{heur}(b_a^z)$.
3. The denominator of $\text{heur}(b_a^z)$.

Each component is estimated separately using independent samples:

1. **Estimating** $P(z|b, a)$: Sample $s_1 \sim b(s)$ and compute:

$$\hat{P}(z|b, a) = \mathbb{E}_{s_1 \sim b(s)} \sum_{s'} T(s_1, a, s')O(s', a, z). \tag{8.15}$$

2. **Estimating the numerator of heur**(b_a^z): Sample $s_2 \sim b(s)$ and compute:

$$\hat{N} = \mathbb{E}_{s_2 \sim b(s)} \sum_s T(s_2, a, s)O(s, a, z)\text{heur}(s). \tag{8.16}$$

3. **Estimating the denominator of heur**(b_a^z): Sample $s_3 \sim b(s)$ and compute:

$$\hat{D} = \mathbb{E}_{s_3 \sim b(s)} \sum_{s'} T(s_3, a, s')O(s', a, z). \tag{8.17}$$

Using these independent estimates, the unbiased estimate of $Q_{\text{init}}(b, a)$ is given by:

$$\hat{Q}_{\text{init}}(b,a) = \hat{\mathcal{C}}(b,a) + \sum_{z \in Z} \hat{P}(z|b,a)\frac{\hat{N}}{\hat{D}}. \tag{8.18}$$

$\hat{\mathcal{C}}(b,a)$ can be estimated using all samples as linearity of expectation holds between $\hat{\mathcal{C}}(b,a)$ and the other estimated quantities.

Practical Considerations

Ensuring that s_1, s_2, and s_3 are sampled independently eliminates the statistical dependency that caused bias in the subsampling estimator. However, this approach introduces a trade-off: while the estimate is now unbiased, it comes at the cost of increased variance. Given a fixed sampling budget B, we must distribute our available samples among three separate estimations, thereby increasing variance in each individual estimate.

In our evaluation, we employed the vanilla subsampling estimator, which performed well in practice. However, if an unbiased estimate of $Q_{\text{init}}(b,a)$ is critical, the outlined method provides a principled approach to achieving it.

Bibliography

[1] A. R. Amazon Science. (2022) The quest to deploy autonomous robots within amazon fulfillment centers. [Online]. Available: https://www.amazon.science/latest-news/the-quest-to-deploy-autonomous-robots-within-amazon-fulfillment-centers

[2] R. Bogue, "Domestic robots: Has their time finally come?" *Industrial Robot: An International Journal*, vol. 44, no. 2, pp. 129–136, 2017.

[3] A. R. Amazon Science. (2022) Robin deals with a world where things are changing all around it. [Online]. Available: https://www.amazon.science/latest-news/robin-deals-with-a-world-where-things-are-changing-all-around-it

[4] J. Forlizzi and C. DiSalvo, "Service robots in the domestic environment: a study of the roomba vacuum in the home," in *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, 2006, pp. 258–265.

[5] T. Asafa, T. Afonja, E. Olaniyan, and H. Alade, "Development of a vacuum cleaner robot," *Alexandria engineering journal*, vol. 57, no. 4, pp. 2911–2920, 2018.

[6] K. Myles and M. S. Binseel, "The tactile modality: a review of tactile sensitivity and human tactile interfaces," *US Army Research Laboratory, ARL-TR-4115, Aberdeen Proving Ground, MD*, vol. 21005, p. 5425, 2007.

[7] R. S. Dahiya, G. Metta, M. Valle, and G. Sandini, "Tactile sensing—from humans to humanoids," *IEEE transactions on robotics*, vol. 26, no. 1, pp. 1–20, 2009.

[8] S. Chitta, E. G. Jones, M. Ciocarlie, and K. Hsiao, "Perception, planning, and execution for mobile manipulation in unstructured environments," *IEEE Robotics and Automation Magazine, Special Issue on Mobile Manipulation*, vol. 19, no. 2, pp. 58–71, 2012.

[9] L. Iocchi, J. Ruiz del Solar, and T. v. d. Zant, "Domestic service robots in the real world," 2012.

[10] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[11] E. J. Sondik, *The optimal control of partially observable Markov processes*. Stanford University, 1971.

[12] A. W. Drake, "Observation of a markov process through a noisy channel," Ph.D. dissertation, Massachusetts Institute of Technology, 1962.

[13] R. Zhou and E. A. Hansen, "An improved grid-based approximation algorithm for pomdps," in *IJCAI*, vol. 1. Citeseer, 2001, pp. 707–716.

[14] M. S. Saleem, R. Veerapaneni, and M. Likhachev, "Preprocessing-based planning for utilizing contacts in semi-structured high-precision insertion tasks," *IEEE Robotics and Automation Letters*, 2023.

[15] F. Islam, O. Salzman, and M. Likhachev, "Provable indefinite-horizon real-time planning for repetitive tasks," in *Proceedings of the International Conference on Automated Planning and Scheduling*, 2019.

[16] F. Islam, O. Salzman, A. Agarwal, and M. Likhachev, "Provably constant-time planning and replanning for real-time grasping objects off a conveyor belt," *The International Journal of Robotics Research*, 2021.

[17] M. S. Saleem, R. Veerapaneni, and M. Likhachev, "A pomdp-based hierarchical planning framework for manipulation under pose uncertainty," *arXiv preprint arXiv:2409.18775*, 2024.

[18] M. S. Saleem, R. Veerapaneni, and M. Likhachev, "Lazy heuristic search for solving pomdps with expensive-to-compute belief transitions," *Proceedings of the International Symposium on Combinatorial Search*, 2025.

[19] M. S. Saleem, L. Yuan, and M. Likhachev, "A contact-driven framework for manipulating in the blind," *arXiv preprint arXiv:2510.20177*, 2025.

[20] M. Vincze and G. D. Hager, "Robust image processing and positionbased visual servoing," 2000.

[21] J. Dong and J. Zhang, "A new image-based visual servoing method with velocity direction control," *Journal of the Franklin Institute*, vol. 357, no. 7, pp. 3993–4007, 2020.

[22] E. Malis, F. Chaumette, and S. Boudet, "Positioning a coarse-calibrated camera with respect to an unknown object by 2d 1/2 visual servoing," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, vol. 2. IEEE, 1998, pp. 1352–1359.

[23] C. Nam, J. Lee, S. H. Cheong, B. Y. Cho, and C. Kim, "Fast and resilient manipulation planning for target retrieval in clutter," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3777–3783.

[24] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox, "6-dof grasping for target-driven object manipulation in clutter," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6232–6238.

[25] S. Cheong, B. Y. Cho, J. Lee, J. Lee, D. H. Kim, C. Nam, C.-h. Kim, and S.-k. Park, "Obstacle rearrangement for robotic manipulation in clutter using a deep q-network," *Intelligent Service Robotics*, vol. 14, no. 4, pp. 549–561, 2021.

[26] M. Q. Mohammed, L. C. Kwek, S. C. Chua, A. Al-Dhaqm, S. Nahavandi, T. A. E. Eisa, M. F. Miskon, M. N. Al-Mhiqani, A. Ali, M. Abaker, *et al.*, "Review of learning-based robotic manipulation in cluttered environments," *Sensors*, vol. 22, no. 20, p. 7938, 2022.

[27] A. Kimmel, R. Shome, Z. Littlefield, and K. Bekris, "Fast, anytime motion planning for prehensile manipulation in clutter," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2018, pp. 1–9.

[28] K. Park, J. Kulick, A. Melkozerov, R. A. Vilagrasa, T. S. Lembono, V. Neubauer, A. Minichev, K. Turner, O. Agrigoroaiei, P. Klink, J. Lee, K. Dwivedi, M. Eltayeb, I. Posner, A. Parness, and C. Erdogan, "Vulcan pick: A robotic system for picking targeted objects from fabric pods," 2025. [Online]. Available: https://www.amazon.science/publications/vulcan-pick-a-robotic-system-for-picking-targeted-objects-from-fabric-pods

[29] M. H. Ali, K. Aizat, K. Yerkhan, T. Zhandos, and O. Anuar, "Vision-based robot manipulator for industrial applications," *Procedia computer science*, vol. 133, pp. 205–212, 2018.

[30] S. Li, A. Keipour, K. Jamieson, N. Hudson, C. Swan, and K. Bekris, "Demonstrating large-scale package manipulation via learned metrics of pick success," *arXiv preprint arXiv:2305.10272*, 2023.

[31] L. Shen, J. Su, and X. Zhang, "Review on peg-in-hole insertion technology based on reinforcement learning," in *2023 China Automation Congress (CAC)*. IEEE, 2023, pp. 6688–6695.

[32] S. Li, H. Gong, J. Liu, J. Li, and X. Deng, "Advances in robotic peg-in-hole assembly: A comprehensive review," *Chinese Journal of Mechanical Engineering*, vol. 38, no. 1, p. 196, 2025.

[33] I. F. Jasim, P. W. Plapper, and H. Voos, "Position identification in force-guided robotic peg-in-hole assembly tasks," *Procedia Cirp*, vol. 23, pp. 217–222, 2014.

[34] Y. Jiang, Z. Huang, B. Yang, and W. Yang, "A review of robotic assembly strategies for the full operation procedure: planning, execution and evaluation," *Robotics and Computer-Integrated Manufacturing*, vol. 78, p. 102366, 2022.

[35] Z. Zhu and H. Hu, "Robot learning from demonstration in robotic assembly: A survey," *Robotics*, vol. 7, no. 2, p. 17, 2018.

[36] B. Saund and D. Berenson, "Motion planning for manipulators in unknown environments with contact sensing uncertainty," in *Proceedings of the 2018 International Symposium on Experimental Robotics*. Springer, 2020, pp. 461–474.

[37] B. Saund, S. Choudhury, S. Srinivasa, and D. Berenson, "The blindfolded robot: A bayesian approach to planning with contact feedback," in *The International Symposium of Robotics Research*. Springer, 2019.

[38] B. L. Saund, "Planning and localization using contacts," Ph.D. dissertation, Carnegie Mellon University Pittsburgh, PA, 2017.

[39] B. Saund, S. Choudhury, S. Srinivasa, and D. Berenson, "The blindfolded traveler's problem: A search framework for motion planning with contact estimates," *The International Journal of Robotics Research*, vol. 42, no. 4-5, pp. 289–309, 2023.

[40] M. Meribout, N. A. Takele, O. Derege, N. Rifiki, M. El Khalil, V. Tiwari, and J. Zhong, "Tactile sensors: A review," *Measurement*, vol. 238, p. 115332, 2024.

[41] R. Bhirangi, T. Hellebrekers, C. Majidi, and A. Gupta, "Reskin: versatile, replaceable, lasting tactile skins," *arXiv preprint arXiv:2111.00071*, 2021.

[42] Y. Wan, Y. Wang, and C. F. Guo, "Recent progresses on flexible tactile sensors," *Materials Today Physics*, vol. 1, pp. 61–73, 2017.

[43] M. Y. Cao, S. Laws, and F. R. y Baena, "Six-axis force/torque sensors for robotics applications: A review," *IEEE Sensors Journal*, vol. 21, no. 24, pp. 27 238–27 251, 2021.

[44] A. Wahrburg, J. Bös, K. D. Listmann, F. Dai, B. Matthias, and H. Ding, "Motor-current-based estimation of cartesian contact forces and torques for robotic manipulators and its application to force control," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 879–886, 2017.

[45] A. De Luca and R. Mattone, "Sensorless robot collision detection and hybrid force/motion control," in *Proceedings of the 2005 IEEE international conference on robotics and automation*, 2005.

[46] A. Petrovskaya and O. Khatib, "Global localization of objects via touch," *IEEE Transactions on Robotics*, 2011.

[47] B. Saund, S. Chen, and R. Simmons, "Touch based localization of parts for high precision manufacturing," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*.

[48] A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng, "Bayesian estimation for autonomous object manipulation based on tactile sensors," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE, 2006, pp. 707–714.

[49] J. Yang, H. Liu, F. Sun, and M. Gao, "Object recognition using tactile and image information," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2015, pp. 1746–1751.

[50] S. Suresh, M. Bauza, K.-T. Yu, J. G. Mangelson, A. Rodriguez, and M. Kaess, "Tactile slam: Real-time inference of shape and pose from planar pushing," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 11 322–11 328.

[51] S. Javdani, M. Klingensmith, A. Bagnell, N. Pollard, and S. Srinivasa, "Efficient touch based localization through submodularity," in *2013 IEEE International Conference on Robotics and Automation*, 2013.

[52] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*. IEEE, 1997, pp. 146–151.

[53] Z. Yi, R. Calandra, F. Veiga, H. van Hoof, T. Hermans, Y. Zhang, and J. Peters, "Active tactile object exploration with gaussian processes," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4925–4930.

[54] H. J. Choi and N. Figueroa, "Gaussian process-based active exploration strategies in vision and touch," *arXiv preprint arXiv:2507.05522*, 2025.

[55] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *Journal of machine learning research*, vol. 22, no. 30, pp. 1–82, 2021.

[56] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[57] F. Li, Q. Jiang, W. Quan, R. Song, and Y. Li, "Manipulation skill acquisition for robotic assembly using deep reinforcement learning," in *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2019, pp. 13–18.

[58] N. Vuong, H. Pham, and Q.-C. Pham, "Learning sequences of manipulation primitives for robotic assembly," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4086–4092.

[59] N. J. Cho, S. H. Lee, J. B. Kim, and I. H. Suh, "Learning, improving, and generalizing motor skills for the peg-in-hole tasks based on imitation learning and self-learning," *Applied Sciences*, vol. 10, no. 8, p. 2719, 2020.

[60] H. Nguyen and H. La, "Review of deep reinforcement learning for robot manipulation," in *2019 Third IEEE international conference on robotic computing (IRC)*. IEEE, 2019, pp. 590–595.

[61] B. Fang, S. Jia, D. Guo, M. Xu, S. Wen, and F. Sun, "Survey of imitation learning for robotic manipulation," *International Journal of Intelligent Robotics and Applications*, vol. 3, no. 4, pp. 362–369, 2019.

[62] Y. Hu, F. Lin, P. Sheng, C. Wen, J. You, and Y. Gao, "Data scaling laws in imitation learning for robotic manipulation," *arXiv preprint arXiv:2410.18647*, 2024.

[63] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," *arXiv preprint arXiv:2403.12945*, 2024.

[64] A. Iosifidis and A. Tefas, *Deep learning for robot perception and cognition*. Academic Press, 2022.

[65] J. Piater, S. Jodogne, R. Detry, D. Kraft, N. Krüger, O. Kroemer, and J. Peters, "Learning visual representations for perception-action systems," *The International Journal of Robotics Research*, vol. 30, no. 3, pp. 294–307, 2011.

[66] P. Pastor, M. Kalakrishnan, S. Chitta, E. Theodorou, and S. Schaal, "Skill learning and task outcome prediction for manipulation," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 3828–3834.

[67] J. Peters, J. Kober, K. Mülling, O. Krämer, and G. Neumann, "Towards robot skill learning: From simple skills to table tennis," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2013, pp. 627–631.

[68] Y. Liu, Z. Li, H. Liu, and Z. Kan, "Skill transfer learning for autonomous robots and human–robot cooperation: A survey," *Robotics and Autonomous Systems*, vol. 128, p. 103515, 2020.

[69] S. Hao, Y. Gu, H. Ma, J. Hong, Z. Wang, D. Wang, and Z. Hu, "Reasoning with language model is planning with world model," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 8154–8173.

[70] B. Say, "A unified framework for planning with learned neural network transition models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 6, 2021, pp. 5016–5024.

[71] B. Bonet and H. Geffner, "Solving pomdps: Rtdp-bel vs. point-based algorithms." in *IJCAI*. Pasadena CA, 2009, pp. 1641–1646.

[72] E. A. Hansen and S. Zilberstein, "Lao*: A heuristic search algorithm that finds solutions with loops," *Artificial Intelligence*, vol. 129, no. 1-2, pp. 35–62, 2001.

[73] R. Bellman, "Dynamic programming," *science*, vol. 153, no. 3731, pp. 34–37, 1966.

[74] J. Pineau, G. Gordon, S. Thrun, *et al.*, "Point-based value iteration: An anytime algorithm for pomdps," in *Ijcai*, vol. 3, 2003, pp. 1025–1032.

[75] T. Smith and R. Simmons, "Heuristic search value iteration for pomdps," *arXiv preprint arXiv:1207.4166*, 2012.

[76] T. Smith and R. Simmons, "Point-based pomdp algorithms: Improved analysis and implementation," *arXiv preprint arXiv:1207.1412*, 2012.

[77] A. G. Barto, S. J. Bradtke, and S. P. Singh, "Learning to act using real-time dynamic programming," *Artificial intelligence*, pp. 81–138, 1995.

[78] S. Kim, O. Salzman, and M. Likhachev, "Pomhdp: Search-based belief space planning using multiple heuristics," in *International Conference on Automated Planning and Scheduling*, 2019.

[79] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE transactions on robotics and automation*, 1996.

[80] J. Hill, "Real time control of a robot with a mobile camera," in *9th Int. Symp. on Industrial Robots, 1979*, 1979, pp. 233–246.

[81] D. Kragic, "Visual servoing for manipulation: Robustness and integration issues." 2003.

[82] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, 1992.

[83] G. D. Hager, W.-C. Chang, and A. S. Morse, "Robot hand-eye coordination based on stereo vision," *IEEE Control Systems Magazine*, vol. 15, no. 1, pp. 30–39, 1995.

[84] F. Chaumette and S. Hutchinson, "Visual servo control. ii. advanced approaches [tutorial]," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 109–118, 2007.

[85] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke, "Training deep neural networks for visual servoing," in *2018 IEEE international conference on robotics and automation (ICRA)*, 2018.

[86] J. Xu, Z. Hou, Z. Liu, and H. Qiao, "Compare contact model-based control and contact model-free learning: A survey of robotic peg-in-hole assembly strategies," *arXiv preprint arXiv:1904.05240*, 2019.

[87] T. Inoue, G. De Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[88] J. Xu, Z. Hou, W. Wang, B. Xu, K. Zhang, and K. Chen, "Feedback deep deterministic policy gradient with fuzzy reward for robotic multiple peg-in-hole assembly tasks," *IEEE Transactions on Industrial Informatics*, 2018.

[89] M. Phillips, B. Cohen, S. Chitta, and M. Likhachev, "E-graphs: Bootstrapping planning with experience graphs," in *Proceedings of the International Symposium on Combinatorial Search*, 2012.

[90] T. Uras, S. Koenig, and C. Hernández, "Subgoal graphs for optimal pathfinding in eight-neighbor grids," in *Proceedings of the International Conference on Automated Planning and Scheduling*, 2013.

[91] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, 1996.

[92] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, 1999.

[93] F. Vaussard, J. Fink, V. Bauwens, P. Rétornaz, D. Hamel, P. Dillenbourg, and F. Mondada, "Lessons learned from robotic vacuum cleaners entering the home ecosystem," *Robotics and Autonomous Systems*, vol. 62, no. 3, pp. 376–391, 2014.

[94] I. Ulrich, F. Mondada, and J.-D. Nicoud, "Autonomous vacuum cleaner," *Robotics and autonomous systems*, vol. 19, no. 3-4, pp. 233–245, 1997.

[95] B. Graf, M. Hans, and R. D. Schraft, "Care-o-bot ii—development of a next generation robotic home assistant," *Autonomous robots*, vol. 16, no. 2, pp. 193–205, 2004.

[96] K. Yamazaki, R. Ueda, S. Nozawa, M. Kojima, K. Okada, K. Matsumoto, M. Ishikawa, I. Shimoyama, and M. Inaba, "Home-assistant robot for an aging society," *Proceedings of the IEEE*, vol. 100, no. 8, pp. 2429–2441, 2012.

[97] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of markov decision processes," *Mathematics of operations research*, vol. 12, no. 3, pp. 441–450, 1987.

[98] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of optimal queueing network control," in *Proceedings of IEEE 9th Annual Conference on Structure in Complexity Theory.* IEEE, 1994, pp. 318–322.

[99] D. Kragic, H. I. Christensen, *et al.*, "Survey on visual servoing for manipulation," *Computational Vision and Active Perception Laboratory, Fiskartorpsv*, vol. 15, p. 2002, 2002.

[100] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, 2023.

[101] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv preprint arXiv:1711.00199*, 2017.

[102] M. C. Koval, N. S. Pollard, and S. S. Srinivasa, "Pose estimation for planar contact manipulation with manifold particle filters," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 922–945, 2015.

[103] S. Pai, T. Chen, M. Tippur, E. Adelson, A. Gupta, and P. Agrawal, "Tactofind: A tactile only system for object retrieval," in *2023 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, 2023, pp. 8025–8032.

[104] N. Jamali, C. Ciliberto, L. Rosasco, and L. Natale, "Active perception: Building objects' models using tactile exploration," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids).* IEEE, 2016, pp. 179–185.

[105] J. Xu, S. Song, and M. Ciocarlie, "Tandem: Learning joint exploration and decision making with tactile sensors," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 391–10 398, 2022.

[106] S. C. Kang, Y. K. Hwang, M. S. Kim, C.-W. Lee, and K.-I. Lee, "A compliant motion control for insertion of complex shaped objects using contact," in *Proceedings of International Conference on Robotics and Automation*, vol. 1. IEEE, 1997, pp. 841–846.

[107] S.-k. Yun, "Compliant manipulation for peg-in-hole: Is passive compliance a key to learn contact motion?" in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 1647–1652.

[108] T. Lozano-Perez, M. T. Mason, and R. H. Taylor, "Automatic synthesis of fine-motion strategies for robots," *The International Journal of Robotics Research*, vol. 3, no. 1, pp. 3–24, 1984.

[109] S. Cao and J. Xiao, "On efficient and flexible autonomous robotic insertion assembly in the presence of uncertainty," *IEEE Robotics and Automation Letters*, 2024.

[110] C. C. Beltran-Hernandez, D. Petit, I. G. Ramirez-Alpizar, and K. Harada, "Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach," *Applied Sciences*, vol. 10, no. 19, p. 6923, 2020.

[111] S. Gubbi, S. Kolathaya, and B. Amrutur, "Imitation learning for high precision peg-in-hole tasks," in *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*. IEEE, 2020, pp. 368–372.

[112] J. Satia and R. Lave, "Markovian decision processes with probabilistic observation of states," *Management Science*, vol. 20, no. 1, pp. 1–13, 1973.

[113] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa, "Online planning algorithms for pomdps," *Journal of Artificial Intelligence Research*, vol. 32, pp. 663–704, 2008.

[114] H. Kurniawati, D. Hsu, and W. S. Lee, "Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces." in *Robotics: Science and systems*, vol. 2008. Citeseer, 2008.

[115] G. Shani, J. Pineau, and R. Kaplow, "A survey of point-based pomdp solvers," *Autonomous Agents and Multi-Agent Systems*, vol. 27, pp. 1–51, 2013.

[116] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," *Advances in neural information processing systems*, vol. 23, 2010.

[117] J. Lee, G.-H. Kim, P. Poupart, and K.-E. Kim, "Monte-carlo tree search for constrained pomdps," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[118] H. Kurniawati and V. Yadav, "An online pomdp solver for uncertainty planning in dynamic environment," in *Robotics Research: The 16th International Symposium ISRR*. Springer, 2016, pp. 611–629.

[119] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "Despot: Online pomdp planning with regularization," *Advances in neural information processing systems*, vol. 26, 2013.

[120] B. Cohen, M. Phillips, and M. Likhachev, "Planning single-arm manipulations with n-arm robots," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 6, no. 1, 2015, pp. 226–227.

[121] C. Dellin and S. Srinivasa, "A unifying formalism for shortest path problems with expensive edge evaluations via lazy best-first search over paths with edge selectors," in *Proceedings of the international conference on automated planning and scheduling*, vol. 26, 2016, pp. 459–467.

[122] A. Mandalika, O. Salzman, and S. Srinivasa, "Lazy receding horizon a* for efficient path planning in graphs with expensive-to-evaluate edges," in *Proceedings of the international conference on automated planning and scheduling*, vol. 28, 2018, pp. 476–484.

[123] M. Bhardwaj, S. Choudhury, B. Boots, and S. Srinivasa, "Leveraging experience in lazy search," *Autonomous Robots*, vol. 45, no. 7, pp. 979–996, 2021.

[124] A. Mandalika, S. Choudhury, O. Salzman, and S. Srinivasa, "Generalized lazy search for robot motion planning: Interleaving search and edge evaluation via event-based toggles," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, 2019, pp. 745–753.

[125] S. Ross, B. Chaib-Draa, *et al.*, "Aems: An anytime online search algorithm for approximate policy refinement in large pomdps." in *IJCAI*, 2007, pp. 2592–2598.

[126] R. Veerapaneni, M. S. Saleem, and M. Likhachev, "Learning local heuristics for search-based navigation planning," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 33, 2023, pp. 634–638.

[127] T. Takahashi, H. Sun, D. Tian, and Y. Wang, "Learning heuristic functions for mobile robot path planning using deep neural networks," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, 2019, pp. 764–772.

[128] M. T. Spaan, "Cooperative active perception using pomdps," in *AAAI 2008 workshop on advancements in POMDP solvers*, 2008.

[129] M. Hauskrecht, "Value-function approximations for partially observable markov decision processes," *Journal of artificial intelligence research*, vol. 13, pp. 33–94, 2000.

[130] H. Warnquist, J. Kvarnström, and P. Doherty, "Iterative bounding lao," in *ECAI 2010*. IOS Press, 2010, pp. 341–346.

[131] N. J. Nilsson, *Principles of artificial intelligence*. Morgan Kaufmann, 2014.

[132] R. Washington, "Bi-pomdp: Bounded, incremental partially-observable markov-model planning," in *European Conference on Planning*. Springer, 1997, pp. 440–451.

[133] A. Schneider, J. Sturm, C. Stachniss, M. Reisert, H. Burkhardt, and W. Burgard, "Object identification with tactile sensors using bag-of-features," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009.

[134] M. Comi, Y. Lin, A. Church, A. Tonioni, L. Aitchison, and N. F. Lepora, "Touchsdf: A deepsdf approach for 3d shape reconstruction using vision-based tactile sensing," *IEEE Robotics and Automation Letters*, 2024.

[135] P. Mittendorfer, E. Yoshida, and G. Cheng, "Realizing whole-body tactile interactions with a self-organizing, multi-modal artificial skin on a humanoid robot," *Advanced Robotics*, 2015.

[136] L. Manuelli and R. Tedrake, "Localizing external contact using proprioceptive sensors: The contact particle filter," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[137] A. Vemula, Y. Oza, J. A. Bagnell, and M. Likhachev, "Planning and execution using inaccurate models with provable guarantees," *arXiv preprint arXiv:2003.04394*, 2020.

[138] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Transactions on Robotics*, 2017.

[139] A. Zwiener, R. Hanten, C. Schulz, and A. Zell, "Armcl: Arm contact point localization via monte carlo localization," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019.

[140] T. Pang, J. Umenberger, and R. Tedrake, "Identifying external contacts from joint torque measurements on serial robotic arms and its limitations," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[141] D. Popov and A. Klimchik, "Transfer learning for collision localization in collaborative robotics," in *Proceedings of the 3rd International Conference on Applications of Intelligent Systems*, 2020.

[142] J. Liang and O. Kroemer, "Contact localization for robot arms in motion without torque sensing," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[143] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool, "Repaint: Inpainting using denoising diffusion probabilistic models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022.

[144] R. Suvorov, E. Logacheva, A. Mashikhin, A. Remizova, A. Ashukha, A. Silvestrov, N. Kong, H. Goka, K. Park, and V. Lempitsky, "Resolution-robust large mask inpainting with fourier convolutions," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2022.

[145] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022.

[146] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019.

[147] A. Dai, C. Ruizhongtai Qi, and M. Nießner, "Shape completion using 3d-encoder-predictor cnns and shape synthesis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[148] L. Zhou, Y. Du, and J. Wu, "3d shape generation and completion through point-voxel diffusion," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[149] J. Zhang, X. Chen, Z. Cai, L. Pan, H. Zhao, S. Yi, C. K. Yeo, B. Dai, and C. C. Loy, "Unsupervised 3d shape completion through gan inversion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021.

[150] E. Weber, A. Holynski, V. Jampani, S. Saxena, N. Snavely, A. Kar, and A. Kanazawa, "Nerfiller: Completing scenes via generative 3d inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[151] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[152] A.-Q. Cao and R. De Charette, "Monoscene: Monocular 3d semantic scene completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[153] Y. Huang, W. Zheng, B. Zhang, J. Zhou, and J. Lu, "Selfocc: Self-supervised vision-based 3d occupancy prediction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024.

[154] R. Cheng, C. Agia, Y. Ren, X. Li, and L. Bingbing, "S3cnet: A sparse semantic scene completion network for lidar point clouds," in *Conference on Robot Learning*. PMLR, 2021.

[155] Z. Xia, Y. Liu, X. Li, X. Zhu, Y. Ma, Y. Li, Y. Hou, and Y. Qiao, "Scpnet: Semantic scene completion on point cloud," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023.

[156] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.

[157] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," *arXiv preprint arXiv:2307.01952*, 2023.

[158] D. Park, A. Kapusta, J. Hawke, and C. C. Kemp, "Interleaving planning and control for efficient haptically-guided reaching in unknown environments," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014.

[159] T. Bhattacharjee, P. Grice, A. Kapusta, M. Killpack, D. Park, and C. Kemp, "A robotic system for reaching in dense clutter that integrates model predictive control, learning, haptic mapping, and planning." Georgia Institute of Technology, 2014.

[160] B. J. Cohen, S. Chitta, and M. Likhachev, "Search-based planning for manipulation with motion primitives," in *2010 IEEE international conference on robotics and automation*. IEEE, 2010.

[161] Z. Xia, X. Pan, S. Song, E. Li, and G. Huang, "Vision transformer with deformable attention," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022.

[162] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, "Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models," *Machine Intelligence Research*, 2025.

[163] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, 2021.

[164] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI conference on artificial intelligence*, 2018.

[165] NVIDIA, "Isaac Sim." [Online]. Available: https://github.com/isaac-sim/IsaacSim

[166] A. Xie and C. Finn, "Lifelong robotic reinforcement learning by retaining experiences," in *Conference on Lifelong Learning Agents*. PMLR, 2022, pp. 838–855.

[167] O. Lemon and U. Nehmzow, "The scientific status of mobile robotics: Multi-resolution map-building as a case study," *Robotics and Autonomous Systems*, vol. 24, no. 1-2, pp. 5–15, 1998.

[168] N. Funk, J. Tarrio, S. Papatheodorou, M. Popović, P. F. Alcantarilla, and S. Leutenegger, "Multi-resolution 3d mapping with explicit free space representation for fast and accurate mobile robot motion planning," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3553–3560, 2021.

[169] X. Ma, J. Zhang, F. Xie, and S. Schwertfeger, "Wifi-based global localization in large-scale environments leveraging structural priors from osmag," *arXiv preprint arXiv:2508.10144*, 2025.

[170] J. Yuan, B. Moon, M. Cao, and S. Scherer, "Hierarchical planning for long-horizon multi-target tracking under target motion uncertainty," *IEEE Robotics and Automation Letters*, 2025.

[171] Y. Jia, C. Wang, W. Tang, Z. Wang, and Z. Sun, "Unlocking full exploration potential of ground robots by multiresolution topological mapping," *IEEE Transactions on Industrial Informatics*, 2025.