Thesis Proposal

Robust Manipulation via Contact Sensing

Michael C. Koval

May 18, 2015

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Siddhartha S. Srinivasa, CMU RI
Nancy S. Pollard, CMU RI
Geoffrey J. Gordon, CMU MLD
Tomás Lozano-Pérez, MIT CSAIL

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

Abstract

Humans effortlessly manipulate objects in cluttered and uncertain environments. In contrast, most robotic manipulators are limited to carefully engineered environments, e.g. factories, to circumvent the difficulty of manipulation under uncertainty. Contact sensors can provide robots with with the feedback vital to addressing this limitation. However, there are three principal challenges to using contact sensing. First, contact is inherently discontinuous. Second, contact sensors only provide rich information while in contact. Third, planning for contact requires reasoning about the physics of interaction.

This thesis proposes a framework for using real-time feedback from contact sensors to reliably manipulate objects under uncertainty. Our framework formulates manipulation as a partially observable Markov decision process. We exploit the structure of the problem to address the challenges of contact and find a policy that reliably achieves a goal.

We show that the optimal policy naturally decouples into pre- and post-contact stages. We present an algorithm that leverages this insight to reuse one post-contact policy across multiple problem instances. Then, during execution, we exploit the lower-dimensional structure of contact to estimate the pose of the object. This algorithm generates policies that succeed even under uncertainty, but makes several assumptions.

In particular, the algorithm (1) requires discretizing the state space at a fixed resolution, (2) ignores kinematic constraints, and (3) assumes perfect proprioception. We propose to relax all three of these assumptions. First, we propose to plan without constructing a fixed discretization of the state space. Second, we propose to consider kinematic constraints during planning by using a powerful heuristic to guide our exploration. Third, we propose to model proprioceptive error by compactly representing belief states using a factored model.

Finally, we evaluate the efficacy of our framework by performing household manipulation tasks both in simulation and on a real robot. We expect that our framework will increase the success rate and improve task performance when compared to algorithms that do not reason about contact sensing during planning.

Contents

1
Introduction

Humans effortlessly manipulate their environment. We are capable of grasping our glasses from our nightstand in the dark, blindly plugging a cable into the back of our computer, and grasping our coffee mug without glancing away from our computer monitor. In the process, we fearlessly push, pull, and slide objects as necessary to complete the task. We use proprioception and tactile sensing to manipulate with confidence despite the variety and complexity of these tasks.

In contrast, most robotic manipulators perform only *pick-and-place* actions in factories. First, the robot perceives an object and chooses a grasp. Then, it uses a *motion planner* to generate a collision-free trajectory to the grasp. Finally, the robot closes its fingers and—hopefully—achieves the desired grasp. This approach ignores the fact that manipulation is a noisy physical process.

Pick-and-place works in industrial applications because the robot has clear access to the target object, is meticulously calibrated, and operates in an environment specifically engineered for a robot to autonomously perform its task. For example, industrial robots frequently are equipped with task-specific end-effectors, track visual fiducials, rely on off-board perception systems, and use fixtures to precisely position the target object.

In many domains it is not possible to engineer the environment in these ways. Robots like HERB [Srinivasa et al., 2012], Andy [Bagnell et al., 2012], and Robonaut 2 [Diftler et al., 2011] operate in environments designed for humans (fig. 1.1). Human environments are cluttered and fundamentally *uncertain*. These robots rely on on-board perception, suffer from occlusion, and rarely have a complete model of their environment (fig. 1.2a). Additionally, robots designed for human environments typically use low-gain, inaccurate, controllers to safely work in proximity to humans (fig. 1.2b). As a result, open-loop actions are brittle and often fail.

Contact sensors provide vital feedback that can be used to reduce

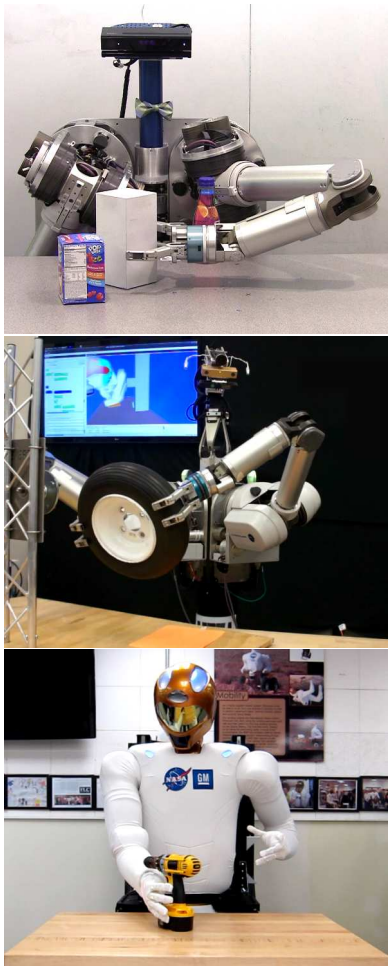


Figure 1.1: HERB (top), Andy (middle), and Robonaut 2 (bottom) manipulating objects in unstructured environments. All three of these robots rely on on-board sensing to cope with uncertainty.

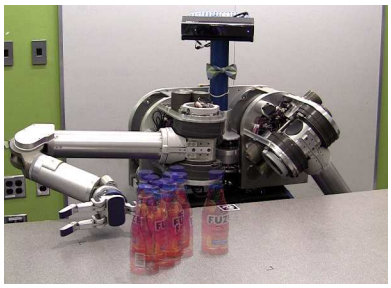
uncertainty. Contact is intimately linked to manipulation: a robot can only apply force to an object while in contact with it. Vision and depth sensors must infer contact from object pose and motion. This is challenging in manipulation because the hand often occludes the object of interest. In contrast, contact sensors directly observe interaction with the object and are unaffected by occlusion.

However, contact sensors have two key limitations. First, contact sensors do not directly estimate object pose. Second, contact sensors provide little information while out of contact with an object. As a result of these limitations, contact sensors are often neglected as a source of real-time feedback during manipulation.

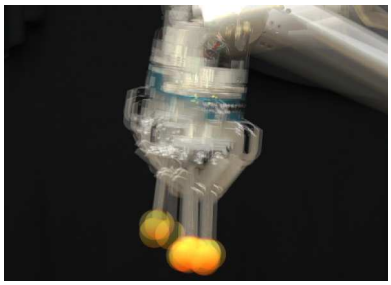
This thesis proposes a framework for using real-time feedback from contact sensors to reliably manipulate objects under uncertainty.

To address this problem in depth, this thesis focuses on the manipulation of a single object using feedback exclusively from proprioception and contact sensors. We assume that the goal can be expressed in configuration space and that some—possibly uncertain—model of the robot, object, and environment are available.

We do not specifically consider planning multi-stage tasks, but anticipate that our framework could be used as a robust primitive action in multi-stage planning algorithms. We also do not consider feedback from other sensing modalities; e.g. vision. However, if a probabilistic model of the additional sensors is available, it is relatively straightforward to incorporate their observation into our state estimate at runtime.



(a) Perceptural Uncertainty



(b) Proprioceptive Uncertainty

Figure 1.2: Robots suffer form uncertainty in both (a) perception and (b) proprioception. Bottom figure courtesy of Klingensmith et al. [2013].

1.1 Challenges with Contact Sensing for Manipulation

There are three principal challenges that make it difficult to use contact sensing for closed-loop manipulation. We survey them here.

Challenge 1: Contact is Inherently Discontinuous

Contact is binary: two rigid bodies are either in contact or they are not. If they are in contact, then the contact may transmit force. If not—even if the bodies are infinitesimally close together—no force is transmitted. As a result, changes in contact are inherently *discontinuous*. The robot may make or break contact by moving an infinitesimal distance. Similarly, an object's motion may change discontinuously.

Contact sensors accurately *discriminate* between contact and no-contact. As a result, contact sensors inherit the discontinuous nature of contact. The output of a contact sensor changes discontinuously when the robot makes or breaks contact with an object.

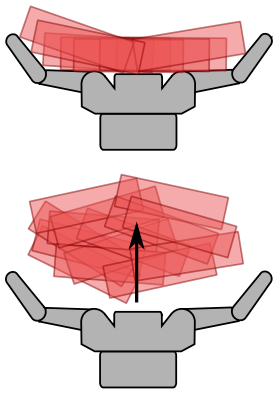
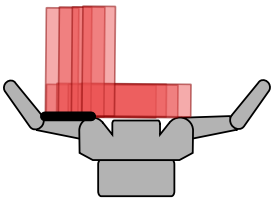


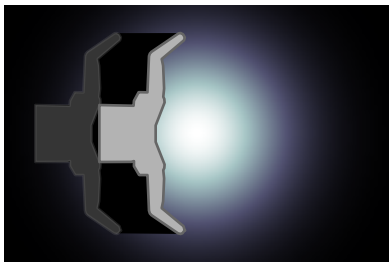
Figure 1.3: Pushing through a produces a discontinuous belief state where the pose of the object is constrained to a line.

Discontinuities introduced by contact accumulate over time into discontinuous belief states. Figure 1.3 shows an example of a pushing action producing a belief state that is constrained to a line. Observations from contact sensors introduce additional discontinuities into the belief state. Figure 1.4 shows an example where receiving a sequence of (a) contact or (b) no-contact observations produces a discontinuous belief state. In fig. 1.4b, the initial belief state was a Gaussian distribution centered in the figure. The posterior belief state assigns zero-probability to the swept volume of the hand because any object in this region would have generated a contact observation.

Planning in this domain is challenging. An infinitesimal change in a policy may result in a discontinuous change in its quality. Policies that exhibit this property are likely to fail when noisily executed on a real system. Our goal is to generate robust policies that succeed despite uncertainty during execution. Additionally, the presence of discontinuous belief states makes it challenging to use planning techniques that rely on discretization or require a compact, parametric representation of belief states.



(a) Contact Observation



(b) No-Contact Observation

Figure 1.4: Receiving (a) contact or (b) no-contact observations may produce a discontinuous belief state.

Challenge 2: Contact Sensors Require Active Sensing

Contact sensors provide rich information while in contact with an object, but little information otherwise. For example, a contact sensor may return a contact position and normal vector during contact. We can use this information to partially infer the object's pose. This is not possible while out of contact with the sensor: we can only infer that the object is not touching the sensor.

Fully utilizing contact sensors requires *active sensing*. The robot must take *information-gathering actions* to force the object into contact with its sensors (fig. 1.5). Information-gathering actions may not directly work towards achieving the goal. Instead, they generate vital observations that reduce uncertainty. Using these observations the robot can robustly achieve the goal.

Synthesizing information-gathering actions requires reasoning about uncertainty during planning. Planning under uncertainty requires planning in *belief space*, the infinite-dimensional space of probability distributions over state. Belief space planning is computationally expensive even in small, discrete domains.

Belief space planning is particularly hard for manipulation. First, manipulation is continuous and high-dimensional. Second, the robot requires multiple observations from contact sensors to infer the full state of the environment. The planner must generate a sophisticated policy that seeks out and combines information from multiple observations to achieves success.

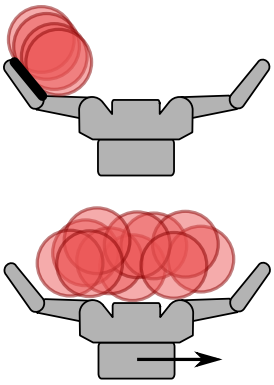


Figure 1.5: The robot moves right to force the bottle into contact with a sensor on its fingertip. This action localizes the bottle and allows the robot to achieve the goal.

These challenges are not unique to contact sensing. Different sensing modalities require differing amounts of information-gathering (fig. 1.6). Long range sensors, like LIDAR (left), are equally accurate throughout the robot's entire workspace. Near-touch sensors (middle) only observe nearby objects. Contact sensors (right) take this to the extreme by only observing objects touching the sensor. We focus on contact sensing in this thesis, but believe that our framework is applicable to other sensing modalities.

Figure 1.6: Different sensing modalities have different maximum ranges. LIDAR (left) has a long range, near-touch sensors (middle) have a short range, and contact sensors (right) have zero range.

Challenge 3: Manipulation is a Physical Process

Making contact with an object also allows the robot to apply force to it. This force may push the object out of the way or—if the object is heavy—significantly affect the motion of the robot. The outcome of the interaction is governed by physics.

Physics can be thought of as a *non-holonomic constraint* on the evolution of the environment's state. The presence of this constraint means that it is difficult to solve the *two-point boundary value problem* in state space; i.e to find a sequence of actions that moves the environment from an initial state to a desired final state. Planning in this type of domain is challenging even when state is fully observed. Most motion planning algorithms gain efficiency by performing a bi-directional search [Kuffner and LaValle, 2000] or building a graph-like structure in configuration space [Kavraki et al., 1996, Karaman and Frazzoli, 2011, Gammell et al., 2015], both of which require solving the two-point boundary value problem.

Additionally, planning with physics requires a *physics model* that can predict the outcome of the robot's actions. Physics models suffer from two key problems. First, simulating physics is computationally expensive and dominates planning time (table 1.1). Second, physics models depend on a variety of physical properties; e.g. those listed in table 1.2. We rarely know the value of these properties in practice. As a result, every action we take introduces uncertainty into the state of the environment.

Operation	Time (ms)	Percent
Physics Model	130.37	80.47%
Sensor Model	29.28	18.07%
Other	2.37	1.46%

Table 1.1: Time spent performing each operation in Koval et al. [2015b]. Times are reported for one update of the filter.

Parameter	Dim.	Units
Center of Mass	3	m
Mass	1	kg
Inertia Tensor	6	$kg \cdot m^2$
Friction Coeffient	1	–
Restitution Coefficient	1	–
Geometry	∞	–

Table 1.2: Properties necessary to simulate a rigid body in the DART [unk, 2013] physics simulator.

1.2 *Key Insights to Our Approach*

The general problem of planning under control and sensing uncertainty is intractable. In this section, we identify the structure unique to manipulation and contact sensing that allows us to plan efficiently.

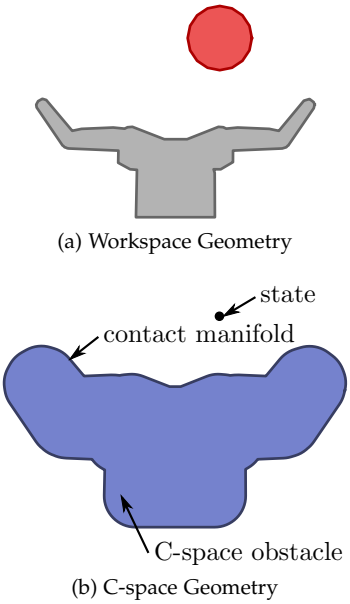
Insight 1: Contact Constrains State to a Manifold

Contact is a constraint on object pose. The configuration space of an object can be partitioned into free space (no contact), an invalid

region (penetrating contact), and a lower-dimensional *contact manifold* (non-penetrating contact). See fig. 1.7 for a graphical depiction of the contact manifold for a radially symmetric object and fig. 1.8 for an asymmetric object.

The lower-dimensional nature of the contact manifold introduces the discontinuity described in challenge 1. We can exploit the structure of the contact manifold to efficiently plan. If we receive a contact observation, then we can infer that an object lies on the lower-dimensional contact manifold of the sensor. This observation eliminates one dimension of uncertainty. Each independent contact further reduces the dimensionality of unobserved state by one.

In our completed work, we sample explicitly from the contact manifold to efficiently track object pose under uncertainty (section 4.1). We also explicitly discretize the contact manifold to capture the discontinuous nature of contact in a relatively coarse discretization of the state space (section 4.2). We propose to use the same structure without computing a coarse, a priori discretization of the state space in section 5.1.



(a) Workspace Geometry

(b) C-space Geometry

Figure 1.7: (a) Decomposition of state space into free space, invalid states, and the contact manifold. (b) The contact manifold is a lower-dimensional structure embedded in configuration space of the object.

Insight 2: Contact Decouples the Optimal Policy

Observing contact implies that the true state of the environment lies on the contact manifold. As a result, we can infer that any hypothesized state that is not on the contact manifold is incorrect. A contact observation can be thought of as a "funnel" that collapses a large set of initial belief states into a smaller set of belief states, each with support restricted to the contact manifold.

This funneling behavior partially decouples the optimal pre-contact policy from the optimal post-contact policy. We expect scenarios that result in the same contact observation to also be solved by similar post-contact policies. This allows us to plan complex policies in belief space without requiring an intractable amount of computation (challenge 2). We have already shown that it is possible to pre-compute a post-contact policy that performs well across a diverse set of scenarios (section 4.2). We propose to extend this decomposition to continuous state spaces (section 5.1) and robots with uncertain proprioception (section 5.3).

In addition to simplifying the post-contact policy, we also use our knowledge of physics to efficiently compute a pre-contact policy. The robot can only interact with an object while in contact it. Therefore, we can trivially solve the two-point boundary value problem when the robot is moving through free space (challenge 3). This is a powerful heuristic for finding the optimal pre-contact policy that we exploited in prior work and propose to extend to robots with

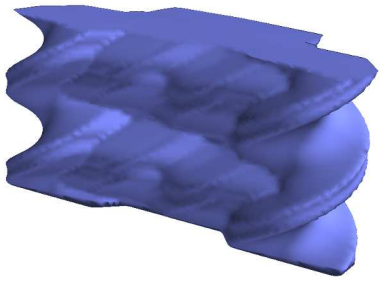


Figure 1.8: For the planar manipulation of non-radially symmetric objects, such as the box shown in fig. 1.3, the contact manifold is a two dimensional structure embedded in $SE(2)$.

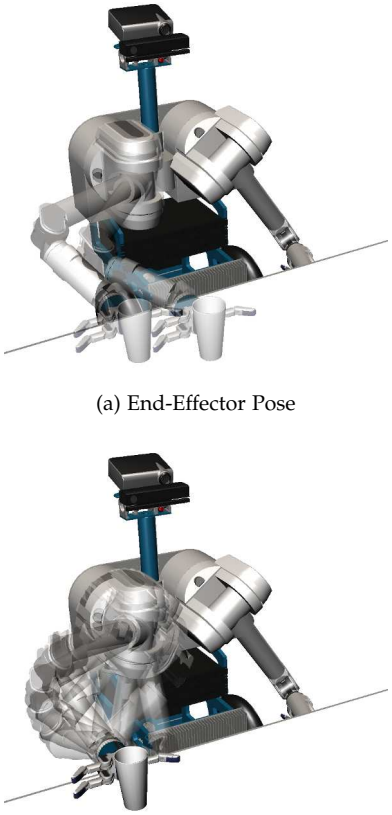
kinematic constraints (section 5.2).

Insight 3: Manipulation Occurs in a Lower-Dimensional Space

Finally, we note that the motion of an object often does not depend on the full state of the environment. Instead, we can write the motion of the object as a function of only the pose and motion of the robot's links that are in contact with it. Under more restrictive assumptions, we can consider only the pose of the object relative to the robot's end-effector (fig. 1.9a).

Solving this lower-dimensional problem may be significantly easier than solving the full problem, i.e. may require fewer evaluations of the physics model (challenge 3). We can project from state space to the lower-dimensional subspace using the robot's forward kinematics. Similarly, we can use inverse kinematics to project back into the full state space (fig. 1.9b). However, this conversion is only optimal if the motion of the robot has no *kinematic constraints*, e.g. joint limits and self-collision.

In prior work, we ignored kinematic constraints and used this structure to plan in the reduced-dimensional state space (section 4.2). We propose to use the lower-dimensional policy to guide a search in the full state space (section 5.2), which may consider arbitrary kinematic constraints. Additionally, we can use a similar approach to cope with tasks where proprioceptive error cannot be projected into the motion of the end-effector (section 5.3).



(a) End-Effector Pose

(b) Kinematic Redundancy

Figure 1.9: Interaction between the object and end-effector is invariant to (a) the end-effector pose and (b) motion in the robot's null space.

1.3 Contributions

The goal of this thesis is to develop a framework that allows robots to use real-time feedback from contact sensors to reliably manipulate objects under uncertainty. We took a first step towards this goal in our completed work:

- The *manifold particle filter* enables robots to estimate the pose of an object using real-time feedback from contact sensors (section 4.1, Koval et al. [2013a,b, 2015b]).
- *Decomposing the optimal policy into pre- and post-contact phases* enables robots to re-use a precomputed post-contact policy for multiple problem instances (section 4.2, Koval et al. [2014, 2015c]).

However, our completed work has several limitations. We propose to overcome three key limitations:

- We plan in a discrete approximation of the continuous state space, which limits the type of actions that the robot can perform. We

propose to *plan directly in the continuous state space* or *incrementally refine our discretization* during planning to expand the planner's capabilities (section 5.1).

- We ignore kinematic constraints, e.g. joint limits and self-collision, during planning. This can cause the planned policy to fail if it encounters kinematic constraints during execution. We propose to *consider kinematic constraints during planning* (section 5.2).
- We ignore proprioception error (e.g. fig. 1.2b) by expressing the goal in the hand frame. This is sufficient for grasping, but cannot express all non-prehensile manipulation tasks. We propose to *consider error in proprioceptive* and plan information-gathering actions that localize the robot relative to the environment (section 5.3).

We plan to evaluate all of the proposed work both in a large suite of simulation experiments. We also plan to perform experiments on HERB [Srinivasa et al., 2012] or Robonaut 2 [Diftler et al., 2011] to verify that the proposed algorithms work on a real robot. Finally, we plan to release an open-source implementation of these algorithms built on the OpenRAVE [Diankov and Kuffner, 2008] or DART [unk, 2013] simulation environments.

2
Background

This thesis builds prior work on non-prehensile manipulation (section 2.1), contact sensing for manipulator control (section 2.2), and contact sensing for state estimation (section 2.3). Similar to some recent work work on motion planning under uncertainty (section 2.4), we formulate we formulate manipulation as a POMDP (section 2.5).

2.1 Open-Loop Pushing Manipulation under Uncertainty

Early work in manipulation addressed the part alignment problem, where a robot plans an open-loop trajectory that reconfigures an object, despite uncertainty in its initial pose [Brokowski et al., 1993, Erdmann and Mason, 1988, Brokowski et al., 1993]. More recently, the same approach has been applied to the problems of grasping [Dogar and Srinivasa, 2010], and rearrangement planning [Dogar and Srinivasa, 2012, Koval et al., 2015a] under the quasistatic assumption [Lynch et al., 1992].

These techniques all consider non-deterministic uncertainty [LaValle and Hutchinson, 1998] in object pose and use worst-case analysis to guarantee success. For example, the push-grasp uses a long, straight-line pushing action to funnel the object into the hand before closing the fingers to achieve a stable grasp [Dogar and Srinivasa, 2010].

Our approach also uses the quasistatic assumption and—when necessary to achieve the goal—generates uncertainty-reducing actions that resemble the push-grasp. However, our approach uses real-time feedback from contact sensors to estimate the pose of the object and achieve the goal. This allows us to achieve the success more quickly and under larger amounts of uncertainty than open-loop strategies.

2.2 Contact Sensing for Manipulator Control

One method of achieving success under uncertainty is to use real-time feedback from contact sensors by directly mapping observa-

tions to actions. Prior work has developed controllers that can locally refine the quality of a grasp [Platt et al., 2010a] or achieve a desired tactile sensor reading [Zhang and Chen, 2000, Li et al., 2013]. These techniques achieve real-time control rates of up to 1.9 kHz [Li et al., 2013] and impressive performance in controlled environments. However—unlike our approach—these algorithms require a high-level planner to analyze the scene and provide a set point to the controller.

It is possible to subvert this problem by directly learning a robust control policy. This has been done by learning a model of expected sensor observations from past experience [Pastor et al., 2011] and using perturbed rollouts to evaluate the effect of uncertainty on each candidate policy [Stulp et al., 2011]. These approaches have been shown to perform well in practice, but policies learned in this way do not easily generalize new tasks or robots. Our approach can be applied to any task or robot for which stochastic transition, observation, and reward functions are available.

2.3 *Contact Sensing for State Estimation*

An alternative use of contact sensing is to estimate the state of the environment during manipulation. Prior work has used a contact sensors to predict grasp stability [Dang et al., 2011] and object identity [Xu et al., 2013, Schneider et al., 2009].

Approaches dating back to the 1970s [Simunovic, 1979] have formulated manipulation under uncertainty as a Bayesian estimation problem. Recently, there has been renewed interest in using contact sensors in the particle to track the pose [Zhang and Trinkle, 2012] and physical parameters [Zhang et al., 2013] of an object being pushed in the plane. Other, closely related work, used a particle filter to track a hybrid discrete-continuous probability distribution over the discrete contact formation [Xiao, 1993] and continuous pose of the object [Meeussen et al., 2007, Gadeyne et al., 2005].

State estimation is an important part of successfully manipulating objects under uncertainty. Our own completed work introduced the manifold particle filter for object pose estimation using contact sensors (section 4.1). The manifold particle filter explicitly samples particles from an approximate representation of the contact manifold to avoid particle starvation during contact (insight 1). However, unlike passive state estimation, our goal is to generate a closed-loop policy that actively achieves a task.

2.4 Motion Planning under Uncertainty

Several motion planning algorithms generate closed-loop policies that achieve a goal under uncertainty. These algorithms include low-level controllers (e.g. those cited in section 2.2), high-level symbolic planners [Smith and Weld, 1998, Hyafil and Bacchus, 2003], and hybrid task planners [Kaelbling and Lozano-Pérez, 2013]. In this thesis, we specifically consider the problem of generating a low-level policy. These policies can be used as primitive actions inside a high level planning framework.

Other work has solved the motion planning under uncertainty problems under the linear-quadratic-Gaussian (LQG) assumptions [Athans, 1971]. In this case, it is efficient to plan by generating and testing candidate trajectories [van den Berg et al., 2010], building a roadmap in state space [Agha-mohammadi et al., 2011, van den Berg et al., 2011], or planning with the maximum-likelihood hypothesis [Platt et al., 2010b]. These techniques have been extended to a variety of application domains. Unfortunately, the belief states encountered during contact manipulation are non-Gaussian and quickly become multi-modal (challenge 3). This precludes us from using techniques that assume that the belief state remains Gaussian.

The idea of planning with the maximum-likelihood hypothesis has also been applied to manipulation [Platt et al., 2011]. This approach uses trajectory optimization to plan a trajectory that either: (1) achieves the goal for a nominal hypothesis or (2) receives observations that invalidate that hypothesis. In the latter case, the algorithm is guaranteed to converge to the goal in a finite number of re-planning steps [Platt. et al., 2012]. Unfortunately, theses techniques aim for feasibility, not optimality. In contrast, our approach optimizes a reward function that drives the robot to quickly achieve the goal.

2.5 Planning for Manipulation under Uncertainty

The work described above solves the general problem of motion planning under uncertainty. In this thesis, we specifically consider planning for manipulation tasks using feedback from contact sensors. Physics-based manipulation under uncertainty is particularly challenging because the observations generated by contact sensors are inherently discontinuous (challenge 1), we must generate information-gathering actions (challenge 2), and we must plan with a physics model (challenge 3).

Early work on robotic assembly used feedback from force sensors to perform fine manipulation [Simunovic, 1979]. A common strategy is to use guarded moves; i.e. move-until-touch actions, to localize

the manipulator relative to the environment [Will and Grossman, 1975]. Guarded moves were constructed by hand [Bolles and Paul, 1973] and, later, synthesized automatically [Lozano-Pèrez et al., 1984]. Recent work has considered the problem of tactile localization [Petrovskaya and Khatib, 2011, Hebert et al., 2013, Javdani et al., 2013], where the robot plans a sequence of guarded move to localize an object.

These techniques split the policy into an information-gathering stage, which attempts to localize the object, followed by a goal-directed stage. An alternative approach is to switch between executing information-gathering and goal-directed trajectories depending upon the amount of uncertainty [Nikandrova et al., 2014]. We propose to eliminate the need for an explicit information-gathering stage by naturally gathering information during execution when doing so is necessary to achieve the goal.

We trade-off between information gathering and goal directed behavior by contact manipulation as a POMDP [Kaelbling et al., 1998]. Hsiao et al. first formulated grasping as a POMDP by decomposing the continuous state space into a discrete set of cells [Hsiao et al., 2007] and, later, by selecting trajectories from a set of candidates [Hsiao et al., 2008]. Both of these approaches assume that the object does not significantly move when touched [Hsiao, 2009]. We consider the case of planar pushing, where motion of the object is critical to achieving the goal.

More recent work has used SARSOP [Kurniawati et al., 2008]—the same point-based POMDP solver we use to to find the post-contact policy—to synthesize an efficient policy that grasps a lug nut under uncertainty [Horowitz and Burdick, 2013]. That work explicitly models the motion of the lug nut and introduces the concept of an *interactivity-based state space* that more densely discretizes states that are near contact.

This approach closely resembles our completed work (section 4.2), which also discretizes the state space and uses SARSOP to find an optimal post-contact policy (section 4.2). We propose to extend this work to deal with discretization errors, cope with kinematic constraints, and proprioceptive uncertainty (chapter 5).

3
Approach

We consider the problem of a robot manipulating an object in a static environment. The robot has a *configuration space* Q and is manipulating an rigid body with configuration space $X = SE(3)$. We represent the joint configuration of the robot and object as a single point in the *state space* $S = Q \times X$.

The state space consists of *immovable obstacles* $S_{\text{obs}} \subseteq S$ and *free space* $S_{\text{free}} = S \setminus S_{\text{obs}}$. We define S_{free} to include its boundary to allow non-penetrating contact between the robot, the object, and immovable obstacles in the environment. The robot's goal is to manipulate the object into a *goal region* $G \subseteq S_{\text{free}}$ defined in this state space.

We consider manipulation to be a stochastic, discrete-time, dynamic system. At each time step, the robot executes an action $a = (\dot{q}, \Delta t) \in A$. An *action* is a velocity $\dot{q} \in \dot{Q}$ that the robot applies for $\Delta t > 0$ seconds. If the robot comes in contact with an immovable object, we assume that the robot's controller will bring it safely to a stop. State evolves according to a stochastic transition function

$$T(s, a, s') = p(s'|s, a)$$

that models the motion of the robot and the physical interaction between the robot and object during contact.

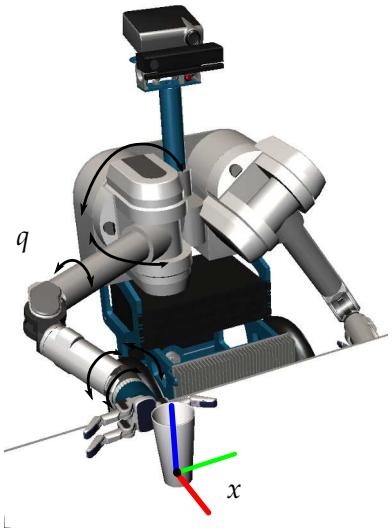
After executing an action $a \in A$, the robot receives an observation $o \in O$. Each observation $o = (o_{\text{q}}, o_{\text{c}}) \in O$ consists of noisy *proprioceptive observation* $o_{\text{q}} \in O_{\text{q}}$ from the robot's internal sensors and a *contact sensor observation* $o_{\text{s}} \in O_{\text{s}}$ from its contact sensors. We assume that observations are generated according to a stochastic *observation model*

$$\Omega(s, o, a) = p(o|s, a) = p(o_{\text{q}}|s, a)p(o_{\text{c}}|s, a),$$

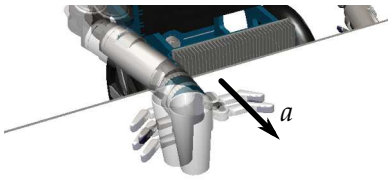
which factors into two terms because proprioceptive and contact sensor observations are conditionally independent given state.

The robot does not know the true state $s \in S$. Instead, it uses the transition and observation models to track a *belief state*

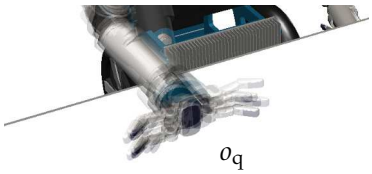
$$b(s_t) = p(s_t|a_{1:t}, o_{1:t})$$



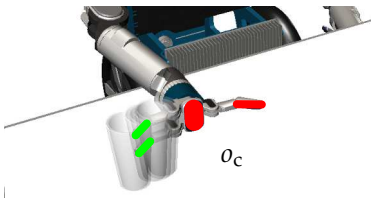
(a) State, S



(b) Action, A



(c) Proprioceptive Observation, O_{q}



(d) Contact Observation, O_{c}

Figure 3.1: (a) State, (b) action, and (c)–(d) observation spaces.

over time. The belief state $b(s_t)$ is a probability distribution over the state s_t given the *history* $(a_1, o_1, \ldots, a_t, o_t)$ of past actions and observations (fig. 3.2). Each belief state $b(s_t) \in \Delta$ is an element of the infinite-dimensional simplex Δ called *belief space*.

Figure 3.2: A belief state is a probability distribution over state s_t given the history up to time t.

The goal of the robot is to find a policy $\pi : \Delta \to A$ over belief space that quickly reaches the goal G. A *policy* chooses which action $\pi[b]$ to execute in belief state $b \in \Delta$. The policy π induces a *value function* $V^\pi : \Delta \to \mathbb{R}$ that encodes the value $V[b]$ of starting in belief state b and following policy π. Given an initial belief state $b(s_0)$, our goal is to find an *optimal policy*

$$\pi^* = \arg\max_\pi V^\pi[b(s_0)]$$

that achieves the highest value $V^{\pi^*}[b(s_0)]$ on belief state $b(s_0)$.

In the remainder of this section, we describe the transition (section 3.1), observation (sections 3.2 and 3.3), and value function (section 3.4) used for manipulation. In the process, we formulate the problem as a *partially observable Markov decision process* (POMDP) [Smallwood and Sondik, 1973, Kaelbling et al., 1998].

3.1 Transition Model

The transition model T encodes the physics of manipulation. We specifically consider the case of quasistatic manipulation of objects resting on a planar support surface. The *quasistatic assumption* states that friction is high enough to neglect acceleration of the object; i.e. the object stops moving as soon as it leaves contact with the robot [Mason, 1986]. This approximation is accurate for the planar manipulation of many household objects [Dogar, 2013].

We implement T using an analytic [Lynch et al., 1992] or numerical [Catto, 2010, Coumanns, 2012, unk, 2013, Smith, 2007] rigid body simulator. We model a physics simulator as a *deterministic* transition function $T^\theta_{\text{det}} : S \times A \to S$ with hyperparameters $\theta \in \Theta$ consisting of *static parameters*. Depending upon the physics model, the parameters θ may include geometry, inertia tensors, and friction coefficients.

In practice, T is stochastic because we do not know the true value of the parameters θ (challenge 3). Instead, we assume that we know a distribution $p(\theta)$ over these parameters. We define the transition model as

$$T(s, a, s') = \int_\theta \delta_{s'}\left(T^\theta_{\text{det}}(s, a)\right) p(\theta) d\theta$$

where $\delta_{s'}(\cdot)$ is the Dirac delta function located at s'. Marginalizing over θ introduces uncertainty into the transition model, while still guaranteeing that s' is feasible [Duff et al., 2010, Duff, 2011]. We

assume that $p(\theta)$ is stationary, which ignores the ability to refine our estimate of θ over time.

3.2 Observation Model for Proprioception

If a robot has has accurate proprioception, then we assume that $o_q = q$ and $p(o_q|q) = \delta_q(o_q)$. In this case the robot's configuration $q \in Q$ is fully observed.

Unfortunately, perfect proprioception is often unavailable. Some robots, such as the Barrett WAM [Salisbury et al., 1988], measure joint angle at the motor instead of the joint. Uncertainty in the transmission between the encoder and joint may introduce significant error in proprioception [Klingensmith et al., 2013, Boots et al., 2014]. Figure 3.3 shows an example of an offset between the measured (solid render) and the actual joint angles (semi-transparent render, right image) on the Barrett WAM. Compliant mechanisms, such as the flexible rubber joints used in the iHY hand [Odhner et al., 2013], have a large number of—and, in some cases, infinite–unobservable degrees of freedom.

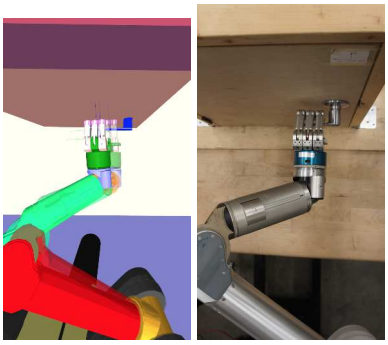


Figure 3.3: Proprioception error on the Barrett WAM arm due to uncertainty in the transmission between the arm's encoders and the joints. Courtesy of Klingensmith et al. [2013].

In general, we expect o_q to differ from q by a time and history dependant offset [Boots et al., 2014]. We do not commit to a specific representation of the proprioceptive observation model $p(o_q|q)$ in this proposal. One simple option is an affine model

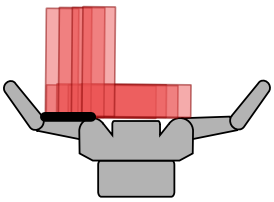
$$o_q = q + q_{\text{offset}} + \epsilon_q$$

that assumes that o_q observes the configuration q corrupted by a static offset $q_{\text{offset}} \in Q$ and Gaussian noise $\epsilon_q \sim \mathcal{N}(0, \Sigma_q)$. In this case, we augment our state space S with the offset q_{offset} and refine our estimate of the offset over time by making contact with the environment. We discuss this problem in section 5.3.

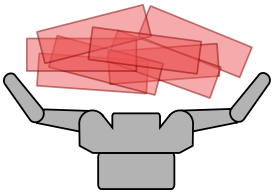
3.3 Observation Model for Contact Sensors

Contact sensors are unique because they provide a wealth of information while in contact, but little information otherwise. We encode this property in the contact sensor observation model $p(o_s|s')$ by partitioning the set of potential contact sensor observations into *contact* $O_c \subseteq O_s$ and *no-contact observations* $O_{nc} = O_s \setminus O_c$.

Similarly, we partition the state space into states $S_c \subseteq S$ where sensor is in contact with geometry (fig. 3.4a) and states $S_{nc} = S \setminus S_c$ where it is not (fig. 3.4b). We assume that contact sensors are *discriminative*; i.e. are unlikely to generate false positives. Formally, we assume that if we receive a contact observation $o \in O_c$, then $\Pr(s' \in S_c) > 1 - \epsilon_c$. This is a common model of contact sensors



(a) Sensor Contact S_c



(b) No Sensor Contact S_{nc}

Figure 3.4: The state space S is partitioned into (a) states in contact with a sensor and (b) those that are not.

that has been successfully used in prior work [Javdani et al., 2013, Hebert et al., 2013, Petrovskaya and Khatib, 2011], including our own completed work in chapter 4.

When $o \in O_c$, the sensor may provide additional information about the state s in the form of the likelihood $p(o_s|s,a)$ over S_c. For example, the resultant measured by a force/torque sensor can be used to constrain the set of feasible contact points. Similarly, observations from a dense array of pressure sensors can be used to estimate the active contact state.

For all of these sensors, the likelihood is uniform over S_{nc} when $o_s \in O_{nc}$. This encodes the important property that observations only provide information about state during contact (challenge 2).

3.4 *Value Function*

We must encode our desire to "quickly reach G" (fig. 3.5) into a value function that we can explicitly optimize. The most obvious choice is to minimize the expected time to reach G with probability $1 - \epsilon_G$. Unfortunately, this objective is undefined if it is not possible to reach G with probability $1 - \epsilon_G$.

The dual to this objective function is to maximize the probability of reaching G subject to a budget of T actions. This objective is also undesirable: there is no incentive for the optimal policy to achieve the goal quickly. In fact, all policies that achieve the goal $1 - \epsilon_G$ within T actions are assigned the same value.

Instead, we encode our goal into a *reward function* $R : S \times A \to \mathbb{R}$ that encourages the robot to reach G. We define

$$R(s,a) = \begin{cases} 0 & : s \in G \\ -\Delta t & : \text{otherwise} \end{cases}$$

by assigning zero reward to R and negative reward to other states.

Our objective is to optimize the sum of expected future reward

$$V^\pi\left[b(s_0)\right] = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right],$$

where the expectation $E[\cdot]$ is taken over the initial state $s_0 \sim b(s_0)$, transitions $s_t \sim p(s_t|s_{t-1}, a_t)$, and observations $o_t \sim p(o_t|s_{t+1})$. Actions are selected by $a_t = \pi[b(s_t)]$. The discount factor $0 \leq \gamma < 1$ trades off between quickly reaching G with low probability and slowly reaching G with high probability.

Defining V^π in terms of R allows us to formulate our problem as a POMDP. As a result, we can build on the rich history of POMDP solvers. Our problem would not be a POMDP if we we chose a value

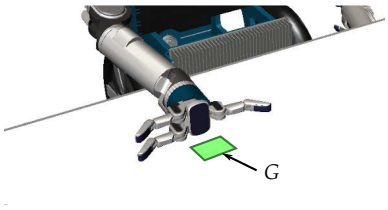


Figure 3.5: The reward function $R(s,a)$ induces a value function $V[b]$ that drives the robot towards the goal $G \subseteq S$.

function that cannot be written as the discounted sum of a reward function defined over $S \times A$.

4
Completed Work

Our goal is to develop a framework that allows robots to use real-time feedback from contact sensors to reliably manipulate objects under uncertainty. This section summarizes our completed work towards this goal.

First, we introduce the *manifold particle filter* as a method of using real-time feedback from contact sensors to estimate the pose of an object relative to the robot's end-effector (section 4.1). We use a set of weighted particles to represent the belief state. Unfortunately, the conventional particle filter performs poorly because state is constrained to a lower dimensional manifold during contact. The manifold particle filter exploits this structure by *explicitly sampling particles from the contact manifold* while observing contact (insight 1).

This is a compilation of the work presented in Koval et al. [2013a,b, 2015b].

Next, we develop an algorithm for efficiently planning under uncertainty when there are no kinematic constraints or proprioceptive error (section 4.2). Our key insight is that *the optimal policy decomposes into pre- and post-contact stages* (insight 2). We compute a post-contact policy once per hand-object pair using a point-based POMDP solver [Kurniawati et al., 2008]. Then, when presented with a new scene, we use an online search to quickly find a pre-contact trajectory that makes contact with the object.

This is a compilation of the work presented in Koval et al. [2014, 2015c].

4.1 Pose Estimation for Contact Manipulation

Planning under uncertainty often requires estimating the current belief state $b(s_t) = p(s_t|a_{1:t}, o_{1:t})$ from a history of actions $a_{1:t} = a_1, \ldots, a_t$ and observations $o_{1:t} = o_1, \ldots, o_t$. We specifically consider the *Bayesian filtering* problem of recursively estimating $b(s_t)$ from our estimate of $b(s_{t-1})$ given that the robot took action a_t and received observation o_t.

To simplify notation, we use s to refer to the object pose relative to the hand x_{rel} and o_t to refer to a contact sensor observation o_c in this section.

The Bayes filter assumes that the system satisfies the *Markov property*. The Markov property states that the current state s_t is a sufficient statistic for all previous actions and observations; i.e.

$s_t \perp (a_{1:t-1}, o_{1:t-1}) | s_{t-1}$ (fig. 4.1). If this property holds, then we can write the recursive *Bayes update* as

$$b(s_t) = \eta \, p(o_t|s_t, a_t) \int_S p(s_t|s_{t-1}, a_t) b(s_{t-1}) ds_{t-1}. \tag{4.1}$$

This section describes an efficient method of implementing eq. (4.1) for the contact manipulation problem using a particle filter. Our key insight is to exploit the structure of the contact manifold (insight 1) to reduce the number of particles required to accurately perform a Bayes update.

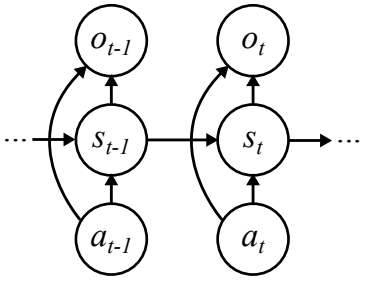


Figure 4.1: Bayes net for the filtering problem. The Markov property states that the future is independent of the past given the current state s_t.

4.1.1 Particle Filter

Recursive filtering requires that we choose a representation of $b(s_t)$ that has the same form as $b(s_{t-1})$ after applying eq. (4.1). The Kalman filter [Kalman, 1960] is optimal when the $b(s_0)$ is Gaussian, the transition model is linear, and observations are corrupted by additive Gaussian white noise. The extended [Kalman, 1960] and unscented [Julier and Uhlmann, 1997] Kalman filters relax the constraint that the system is linear, but still assume that the belief state is Gaussian. As we showed in challenge 1, this is a poor approximation for manipulation.

Instead, we approximate $b(s_t)$ with the distribution

$$b(s_t) \approx \sum_{i=1}^{n_t} w_t^{[i]} \delta_{s_t}(x^{[i]})$$

defined over a set of n_t weighted samples $S_t = \{\langle s_t^{[i]}, w_t^{[i]} \rangle\}_{i=1}^{n_t}$. The samples, along with their weights, are called *particles* and are distributed according to the belief state $b(s_t)$.

We say that S_t is "distributed according to $b(s_t)$" if eq. (4.2) is true for all functions $f : S \to \mathbb{R}$. We abuse notation and denote this by $S_t \sim b(s_t)$.

If this condition is true, we can approximate the expectation of any function $f : S \to \mathbb{R}$ as

$$E_{s_t \sim b(s_t)}[f(s_t)] = \int_S f(s_t) b(s_t) ds_t \approx \sum_{i=1}^{n_t} w_t^{[i]} f\left(s_t^{[i]}\right) \tag{4.2}$$

by replacing the integral over S with a summation over X_t. This includes the reward function $R : S \times A \to \mathbb{R}$ defined in section 3.4.

The *particle filter* uses *importance sampling* to recursively construct S_t from S_{t-1} [Gordon et al., 1993]. Ideally, we would generate S_t by directly sampling from the *target distribution* in eq. (4.1). Unfortunately, this is not straightforward. Instead, we sample from a *proposal distribution* $s_t^{[i]} \sim q(s_t)$ and compute *importance weights* $w_t^{[i]} \sim b(s_t)/q(s_t)$. These weights correct for the mismatch between $q(s_t)$ and $b(s_t)$. Finally, we sample from S_t in proportion to with replacement to normalize the weights.

Algorithm 1 Conventional Particle Filter

Input: action $a_t \in A$ and observation $o_t \in O$
Input: particles $S_{t-1} = \{\langle s_{t-1}^{[i]}, w_{t-1}^{[i]} \rangle\}_{i=1}^{n}$ s.t. $S_{t-1} \sim b(s_{t-1})$
Output: particles $S_t = \{\langle s_t^{[i]}, w_t^{[i]} \rangle\}_{i=1}^{n}$ s.t. $S_t \sim b(s_t)$

```
Ŝ_t ← ∅
for i = 1, ..., |S_{t-1}| do
    s_t^[i] ~ p(s_t^[i] | s_{t-1}^[i], a_t)
    w_t^[i] ← w_{t-1}^[i] p(o_t | s_t^[i], a_t)
    Ŝ_t ← {⟨s_t^[i], w_t^[i]⟩} ∪ Ŝ_t
end for
S_t ← Resample(Ŝ_t)
```

4.1.1.1 Conventional Proposal Distribution

The only restriction on our choice of proposal distribution $q(s_t)$ is that the support of $q(s_t)$ is a superset of the support of $b(s_t)$; i.e. $b(s_t) > 0 \implies q(s_t) > 0$. However, in practice, we must choose $q(s_t)$ such that: (1) we can sample $s_t^{[i]} \sim q(s_t)$ and (2) we can compute the importance weights $w_t^{[i]} = b(s_t)/q(s_t)$.

Conventionally, we choose the proposal distribution

$$q(s_t) = \int_S p(s_t|s_{t-1}, a_t) b(s_{t-1}) ds_{t-1}\,, \tag{4.3}$$

which is the belief state after taking action a_t, but before receiving an observation. The corresponding importance weights are

$$w_t^{[i]} = \frac{b(s_t^{[i]})}{q(s_t^{[i]})} = \eta\, p(o_t|s_t^{[i]}, a_t)\,,$$

which integrate our observation of o_t into the posterior belief state.

We sample from eq. (4.3) by forward-propagating each sample $s_{t-1}^{[i]}$ through the transition model $s_t^{[i]} \sim p(s_t^{[i]}|s_{t-1}^{[i]}, a_t)$. We evaluate $p(o_t|s_t^{[i]})$ using the model described in section 3.3. Algorithm 1 summarizes the *conventional particle filter*.

4.1.1.2 Degeneracy of the Conventional Proposal Distribution

Unfortunately, importance sampling from this proposal distribution performs poorly when using contact sensors. Contact sensors are discontinuous (challenge 1) and only respond to state on a lower-dimensional contact manifold. As a result, the conventional proposal distribution is a poor approximation of the target distribution when receiving contact observations. This results in *particle starvation*, where no particles in S_t agree with o_t with high probability (fig. 4.2).

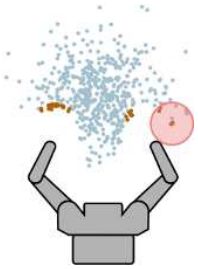


Figure 4.2: Only the small set of particles (dark orange) that are in the swept volume of the sensors generate contact observations. Most particles (light blue) generate no-contact observations and have low weight during contact.

Surprisingly, this causes the *conventional particle filter* (CPF) to *perform worse as the sensor resolution increases*. We illustrate the reason for this unintuitive result in fig. 4.3. As sensor resolution increases (left-to-right), the swept volume of each sensor becomes narrower. As the update frequency increases (top-to-bottom), the distance traveled by the hand between updates decreases, and the swept volume shrinks.

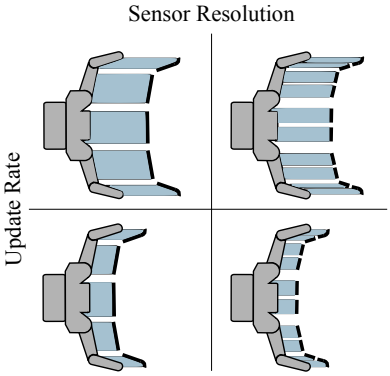


Figure 4.3: Increasing the sensor's resolution (left-to-right) or update rate (top-to-bottom) reduces the swept volume of the sensors. This exacerbates the problem of particle starvation.

As in prior work, we assume that $p(o_t|a_t)$ is finite [Thrun et al., 2000b].

4.1.1.3 Dual Proposal Distribution

Alternatively, we can sample from the *dual proposal distribution*

$$\bar{q}(s_t) = \frac{p(o_t|s_t, a_t)}{p(o_t|a_t)} \tag{4.4}$$

to generate a sample $s_t^{[i]} \sim \bar{q}(s_t)$ that is consistent with the latest observation o_t. Equation (4.4) ignores $b(s_{t-1})$ and generates samples that are likely to produce observation o_t.

The corresponding *dual importance weight* is

$$\bar{w}_t^{[i]} = \frac{b(s_t^{[i]})}{\bar{q}(s_t^{[i]})} = \eta' \int_S p(s_t^{[i]}|s_{t-1}, a_t) b(s_{t-1}) ds_{t-1}$$

with normalization factor η'. We obtained this expression by dividing the target distribution (eq. (4.1)) by the proposal distribution (eq. (4.4)). The dual importance weights incorporates the prior belief state $b(s_{t-1})$ and the most recent action a_t into our estimate of $b(s_t)$.

4.1.2 *Manifold Particle Filter*

The conventional proposal distribution (section 4.1.1.1) performs well when the output of the transition model is more informative than the output of the observation model. Conversely, the dual proposal distribution (section 4.1.1.3) performs best when the observation model is informative. Neither of these cases apply to contact sensors.

Contact sensors provide a wealth of information while in contact, but little information while in free space (challenges 1 and 2). Our key insight is to factor the belief state $b(s_t)$ into

$$b(s_t) = b(s_t \in S_c) b(s_t|S_c) + b(s_t \in S_{nc}) b(s_t|S_{nc})$$

where $b(s_t|S_c)$ is a distribution over set of states S_c in contact with a sensor and $b(s_t|S_{nc})$ is a distribution over the remainder of the state space $S_{nc} = S \setminus S_c$.

The *manifold particle filter* samples in two steps. First, we sample the set S_c or S_{free} according to their probabilities. Then, we sample from a proposal distribution over that set. If $s_t \in S_c$ we choose from the dual proposal distribution $\bar{q}(s_t|S_c)$. Otherwise, we choose

Algorithm 2 Manifold Particle Filter

Input: number of dual particles d_t, mixing rate ϕ
Input: action $a_t \in A$ and observation $o_t \in O$
Input: particles $S_{t-1} = \{\langle s_{t-1}^{[i]}, w_{t-1}^{[i]} \rangle\}_{i=1}^{n}$ s.t. $S_{t-1} \sim b(s_{t-1})$
Output: particles $S_t = \{\langle s_t^{[i]}, w_t^{[i]} \rangle\}_{i=1}^{n}$ s.t. $S_t \sim b(s_t)$

$S_{\text{cpf},t} \leftarrow \text{ConventionalParticleFilter}(S_{t-1}, a_t, o_t)$ ▷ algorithm 1
for $i = 1, \ldots, d_t$ **do**
  $\bar{s}_t^{[i]} \sim p(o_t | s_t^{[i]}, a_t)$ ▷ section 4.1.2.2
  $\bar{w}_t^{[i]} = \int_S p(s_t^{[i]} | s_{t-1}, a_t) b(s_{t-1}) ds_{t-1}$ ▷ section 4.1.2.3
  $\bar{S}_t \leftarrow \{\langle \bar{s}_t^{[i]}, \bar{w}_t^{[i]} \rangle\} \cup \bar{S}_t$
end for
$\hat{S}_t \leftarrow b(s_t \in S_{\text{nc}}) S_{\text{cpf},t} + b(s_t \in S_{\text{c}}) \bar{S}_t$ ▷ section 4.1.2.4
$S_t \leftarrow \text{Resample}(\hat{S}_t)$

the conventional proposal distribution $q(s_t | S_{\text{nc}})$. This combines the advantages of both approaches.

There are three key challenges in implementing the manifold particle filter. First, we must estimate $b(s_t \in S_{\text{c}})$. Second, we must sample from the dual proposal $\bar{s}_t^{[i]} \sim \hat{q}(\bar{s}_t^{[i]} | S_{\text{c}})$ distribution. Finally, we must compute the corresponding importance weight $\bar{w}_t^{[i]}$.

4.1.2.1 Estimating the Probability of Contact

Factoring $b(s_t)$ into contact and no-contact requires an estimate of the probability of contact $b(s_t \in S_{\text{c}})$. We would typically compute

$$b(s_t \in S_{\text{c}}) = \int_{S_{\text{c}}} b(s_t) ds_t$$

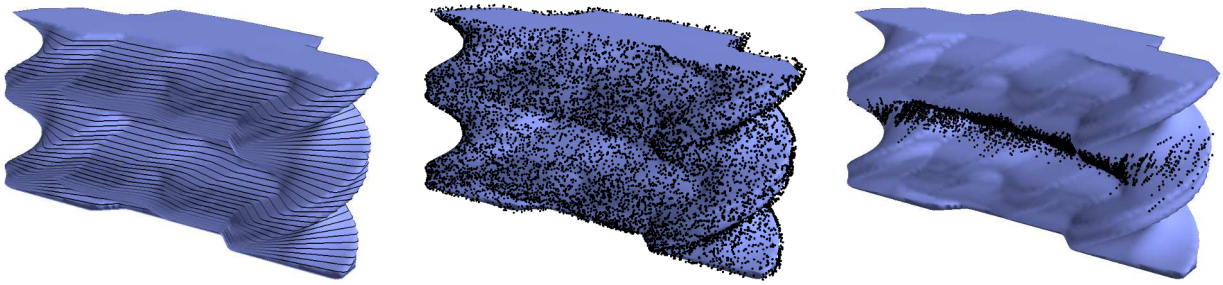
by marginalizing the belief $b(s_t)$ over S_{c}. Unfortunately, approximating this integral requires knowledge of $b(s_t)$: precisely the distribution we are trying to estimate.

Given $b(s_t \in S_{\text{c}})$ we can compute $b(s_t \in S_{\text{nc}}) = 1 - b(s_t \in S_{\text{c}})$.

Instead, we assume $b(s_t \in S_{\text{c}}) = 1$ when $o_t \in O_{\text{c}}$ and $b(s_t \in S_{\text{c}}) = 0$ otherwise. This assumption is correct when contact sensors are perfectly discriminative; i.e. $\epsilon_{\text{c}} = 0$, because $p(o_t | s_t, a_t) = 0$ when $o_t \in O_{\text{c}}$ and $s_t \notin S_{\text{c}}$. This is a good approximation of the true belief dynamics when the contact sensors have a low probability of mis-detecting contact.

4.1.2.2 Sampling from the Dual Proposal Distribution

During contact s_t is constrained to the lower-dimensional contact manifold $S_{\text{c}} \subseteq S$ and we sample from the dual proposal distribution $\bar{q}(s_t | S_{\text{c}})$. To do so, we build an approximation representation of S_{c} and sample from it.



(a) Analytic (b) Rejection-Sampled (c) Trajectory Rollout

Figure 4.4: (a) Analytic, (b) rejection sampling, and (c) trajectory rollout representations of the contact manifold.

In some special cases, e.g. when the hand and object are polygons, we can compute an *analytic representation* of S_c. In this case, we use the Minkowski sum to compute the C-obstacle of the hand in the configuration space of the object with a fixed orientation [Lozano-Pèrez, 1983]. We repeat this computation for many orientations of the object and approximate S_c as the union of the boundaries of these iso-contours (fig. 4.4a). We sample $\bar{s}_t$ by first sampling an orientation, then sampling a position from the corresponding iso-contour.

In general it is not feasible to compute an analytic representation of S_c. In this case, we approximate S_c with a discrete set of samples $\tilde{S}_c \subseteq S_c$ near the manifold (fig. 4.4b). The most straightforward way of sampling from $S_c \subset S$ is to rejection sample from the ambient space S. *Rejection sampling* iteratively samples candidate states $s^{[i]} \sim \text{uniform}(S)$ until it finds a sample $\min_{s \in S_c} ||s - s^{[i]}|| \leq \epsilon_{rs}$ near desired set. The value $\epsilon_{rs} > 0$ is a tolerance parameter.

Ideally we would set $\epsilon_{rs} = 0$ and rejection sample from S_c. However, since S_c has zero measure, there is zero probability of this succeeding.

Rejection sampling attempts to cover all of S_c with uniform density. As a result, many of the samples $\tilde{S}_c$ will be found in regions of S_c that remain low probability during execution. We exploit this structure by concentrating samples in the regions of S_c that we encounter during execution (fig. 4.4c). We do so by performing *trajectory rollouts* from the initial belief $b(s_0)$. Each time we encounter a state $s \in S_c$, we add s to $\tilde{S}_c$. We repeat this process until $\tilde{S}_c$ reaches the desired size. At runtime, we sample from the discrete set $\tilde{S}_c$.

4.1.2.3 Computing Dual Importance Weights

Once we have drawn a sample $\bar{s}_t^{[i]}$ from the dual proposal distribution, we must compute its corresponding importance weight $\bar{w}_t^{[i]}$. Intuitively, the importance weight integrates our prior belief $b(s_{t-1})$ and the effect of taking action a_t into $b(s_t)$ [Thrun et al., 2000a].

We evaluate $w^{[i]}$ by forward propagating each particle $s_{t-1}^{[i]} \in S_{t-1}$ from time $t-1$ to time t using the transition model $s_t^{[i]} \sim p(s_t^{[i]} | s_{t-1}^{[i]}, a_t)$. This set of samples, which we denote by S_{t-1}^{+}, is dis-

tributed according to our belief state after taking action a_t, but before receiving the next observation o_t. Then, we use S^+_{t-1} to approximate the importance weight $\bar{w}^{[i]} = \int_S p(s_t|s_{t-1}, a_t) b(s_{t-1}) ds_{t-1}$ using a density estimation technique [Thrun et al., 2000a].

We use kernel density estimation [Rosenblatt, 1956] to promote S^+_{t-1} into a full-dimensional distribution over S and evaluate $w^{[i]}_t$ using the density estimate over the full space. The belief given by our forward propagated particles $S^+_{t-1} = \langle s^{[i]}_{t-1,+}, w^{[i]}_{t-1,+} \rangle^n_{i=1}$ is

$$b(s_{t-1,+}) \approx \sum_{i=1}^{n} w^{[i]}_{t-1,+} K\left(s_{t-1,+}, s^{[i]}_{t-1,+}\right),$$

where $K : S \times S \to \mathbb{R}$ is a kernel function. This allows us to evaluate $b(s_{t-1,+})$ for the particles S_t that we sampled from the dual proposal distribution.

In practice, we choose $K(\cdot, \cdot)$ to be a Gaussian kernel and select the bandwidth matrix using a multivariate generalization of Silverman's rule of thumb [Silverman, 1981]. Our estimate is effectively restricted to S_c because it is only evaluated on the samples drawn from the dual proposal distribution.

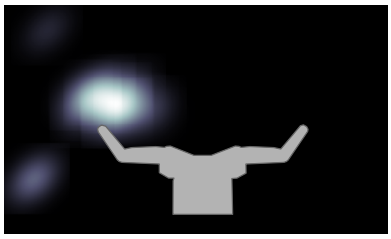


Figure 4.5: Kernel density estimate used to compute dual importance weights.

See Koval et al. [2015b] for a discussion of the impact of this approximation.

4.1.2.4 Mixture Proposal Distribution

Just as how the conventional proposal distribution performs poorly with accurate sensors, the dual proposal distribution performs poorly with observation noise [Thrun et al., 2000a]. The MPF uses the dual proposal distribution to sample from S_c and, as a result, inherits the same weakness.

We use a *mixture proposal distribution* [Thrun et al., 2000a] to mitigate this effect by combining both sampling techniques. Instead of sampling all of the particles from the MPF, we sample some particles $|S_{\text{cpf}}| = n$ from the CPF and the remaining particles $|S_{\text{mpf}}| = d$ from the MPF. The mixture proposal distribution combines the two sets of particles with the weighted sum $(1 - \phi)S_{\text{cpf}} + \phi S_{\text{mpf}}$ with $0 \leq \phi \leq 1$ before resampling.

We seamlessly combine the mixture proposal distribution and the MPF's manifold mixing step, which combines particles $S^{S_{\text{nc}}}_t$ sampled from the conventional proposal distribution with particles $S^{S_c}_t$ sampled from the dual proposal distribution, into a single update. To do

We define the sum $aX + cY$ of the sets of particles $X = \{\langle x^{[i]}, w^{[i]}_x \rangle\}^{n_x}_{i=1}$ and $Y = \{\langle y^{[i]}, w^{[i]}_y \rangle\}^{n_y}_{i=1}$ with non-negative scale factors $a, c \in \mathbb{R}^{\geq 0}$ to be $aX + cY = \{\langle x^{[i]}, a w^{[i]}_x / W_x \rangle\}^{n_x}_{i=0} \cup \{\langle y^{[i]}, c w^{[i]}_y / W_y \rangle\}^{n_y}_{i=0}$. The variables $W_x = \sum^{n_x}_{i=1} w^{[i]}_x$ and $W_y = \sum^{n_y}_{i=1} w^{[i]}_y$ denote the total weight of X and Y, respectively.

so, we rewrite the mixture proposal distribution as

$$\begin{aligned} S_t &= (1-\phi)S_{\text{cpf}} + \phi\left[b(s_t \in S_{\text{c}})S_t^{S_{\text{c}}} + b(s_t \in S_{\text{nc}})S_t^{S_{\text{nc}}}\right] \\ &= (1-\phi)S_{\text{cpf}} + \phi\left[b(s_t \in S_{\text{c}})S_t^{S_{\text{c}}} + b(s_t \in S_{\text{nc}})(S_{\text{cpf}} \cap S_{\text{nc}})\right] \\ &= (1-\phi)(S_{\text{cpf}} \cap S_{\text{c}}) + \left[1 - \phi + \phi\, b(s_t \in S_{\text{nc}})\right](S_{\text{cpf}} \cap S_{\text{nc}}) \\ &\quad + \phi\, b(s_t \in S_{\text{c}})S_t^{S_{\text{c}}} \end{aligned}$$

by partitioning S_{cpf} into the particles $S_{\text{cpf}} \cap S_{\text{c}}$ on the observable contact manifold S_{c} and those $S_{\text{c}} \cap S_{\text{nc}}$ in free space. This factorization is possible because both $S_{\text{cpf}} \cap S_{\text{nc}}$ and $S_t^{S_{\text{nc}}}$ are both generated by sampling from the conventional proposal distribution. This combined update rule can be interpreted as assigning additional weight to the particles $S_{\text{cpf}} \cap S_{\text{nc}}$ to avoid biasing S_t towards S_{c}.

The parameters n and d can be interpreted as the minimum number of samples necessary to cover the high-probability regions of, respectively, the ambient space S_{nc} and the observable contact manifold S_{c}. The *mixing rate* $0 \leq \phi \leq 1$ parameter allows the algorithm to smoothly transition between the CPF ($\phi = 0$) to the MPF ($\phi = 1$). Increasing ϕ provides better performance when transitioning between manifolds, but only at the cost of becoming more sensitive to erroneous observations [Thrun et al., 2000a].

The MPF maintains a single set of particles that span all manifolds. It is not meaningful to identify whether a particular particle was sampled from the "conventional" or "dual" proposal distribution since they are seamlessly mixed as part of the same posterior distribution.

4.1.3 Simulation Experiments

We designed a set of simulation experiments to compare the MPF with the CPF. Based on the particle starvation problem described in section 4.1.1.2, we hypothesize that:

H1. *The MPF will outperform the CPF after contact.*

Increasing the sensor resolution or update rate should make this difference more pronounced because it reduces the swept volume of the sensors. As this happens, the CPF will begin to suffer from particle starvation. Therefore, we hypothesize:

H2. *The CPF will perform worse as sensor resolution increases; the MPF will perform better.*

H3. *The CPF will perform worse as the sensor update rate increases; the MPF will perform better.*

All of the above hypotheses (H1–H3) should be true regardless of which representation of the contact manifold is used by the MPF.

Since AM faithfully represents the continuous manifold, we hypothesize:

H4. *The analytic representation of the contact manifold will perform best.*

Surprisingly, our results suggest that H4 is false: TR outperforms AM despite the fact that it is a sample-based approximation of the true contact manifold. This occurs because TR only represents the region of the contact manifold that can be reached during execution.

Between the sample-based representations, we expect TR to outperform RS. RS attempts to represent the contact manifold at a uniform resolution. In contrast, TR focuses samples on regions of the contact manifold that we are likely to encounter during execution.

Therefore, we hypothesize:

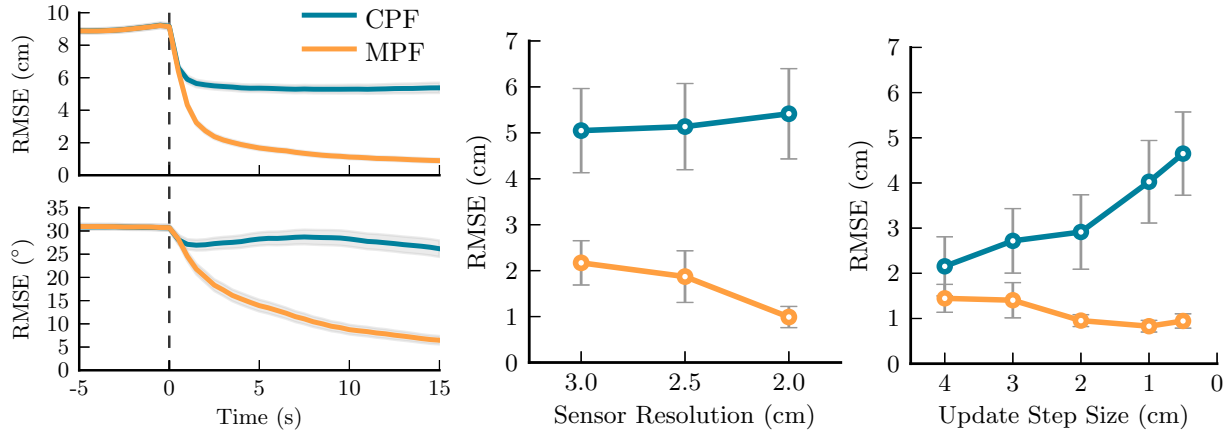
H5. *The trajectory rollout representation will outperform the rejection sampled representation.*

4.1.3.1 Experimental Design

We implemented the CPF and MPF in a custom two-dimensional kinematic simulation environment with polygonal geometry. Each experiment consisted of a simulated BarrettHand pushing a rectangular box in a straight line at a speed of 1 cm/s for 50 cm. The initial belief state was set to $b(s_0) = \mathcal{N}(\bar{s}_0, \Sigma)$ with covariance $\Sigma^{1/2} = \text{diag}[5\text{ cm}, 5\text{ cm}, 20^\circ]$. The mean $\bar{s}_0 = (\bar{x}_0, \bar{y}_0, \bar{\theta}_0)$ was placed a fixed distance $\bar{x}_0 = 20$ cm from the hand and was assigned a random lateral offset $\bar{y}_0 \sim \text{uniform}[-10\text{ cm}, 10\text{ cm}]$ and orientation $\bar{\theta}_0 \sim \text{uniform}[0^\circ, 360^\circ]$ for each trial.

We simulated the motion of the object using a penetration-based quasistatic physics model [Lynch et al., 1992] with a 1 mm step size. Before each step, the finger-object coefficient of friction μ_f and the radius of the object's pressure distribution c were sampled from the Gaussian distributions $\mu_f \sim N(0.5, 0.2^2)$ and $c \sim N(0.05, 0.01^2)$ truncated to enforce $\mu_f, c > 0$.

Binary observations were simulated for each of the hand's sensors, which were uniformly distributed across the front surface of the hand, by computing the intersection of each sensor with the object. Observations were assumed to be perfectly discriminative, but the observation model had a 10% chance of generating an incorrect observation during contact; i.e. an incorrect sensor would fire. These observations were simulated by applying the same observation model to a special "ground truth" particle $s_0^* \sim b(s_0)$.



(a) Estimation Error (b) Spatial Resolution (c) Temporal Resolution

Figure 4.6: Comparison between the CPF and the MPF-AM in simulation. (a) The CPF and MPF perform identically before contact, but the MPF outperforms the CPF post-contact. (b) The MPF improves as spatial sensor resolution increases, whereas the CPF declines in performance. (c) Similarly, the MPF improves and the CPF declines when faced with a faster update frequency. Error bars indicate a 95% confidence interval.

4.1.3.2 Dependent Measures

We evaluate the performance of an estimator by computing the *root mean square error*

$$\mathrm{RMSE}(S_t, s_t^*) = \sqrt{\frac{\sum_{i=1}^{n} (s_t^{[i]} - s_t^*)^2 w_t^{[i]}}{\sum_{i=1}^{n} w_t^{[i]}}}$$

of the particles $S_t = \{\langle s_t^{[i]}, w_t^{[i]} \rangle\}_{i=1}^{n}$ with respect to the true state s_t^* at time t. Instead of combining the position error (measured in centimeters) with the orientation error (measured in degrees), we report separate RMSE values for position and orientation.

4.1.3.3 Conventional vs. Manifold Particle Filter (H1)

We ran the CPF with $n = 100$ particles and the MPF with $n = 100$ conventional particles, $d = 25$ dual particles, and a mixing rate of $\phi = 0.1$. We intentionally chose the same value of n for both algorithms—despite the addition of d dual particles for the MPF—because the dual sampling step adds negligible overhead to the runtime of the algorithm. This means that both the CPF and MPF are tuned to run at approximately the same update rate. The MPF sampled from an analytic representation of the contact manifold that was pre-computed with 1 mm linear and 1.15° angular resolution.

Figure 4.6a shows the performance of both filters averaged over 900 trials. These results show that—as expected—both filters behave similarly before contact ($t \leq 0$) and there was not a significant difference in RMSE. After contact ($t > 0$), the MPF quickly achieves 4.4 cm less RMSE than the CPF.

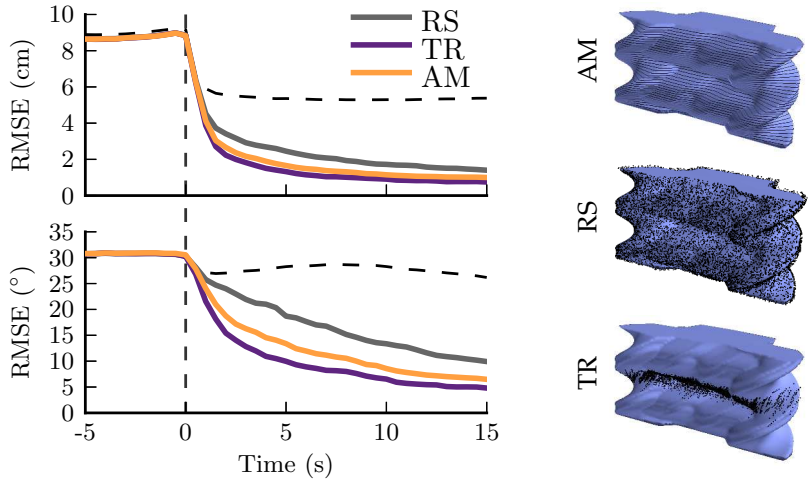


Figure 4.7: Performance of MPF using the rejection-sampled (RS), trajectory-rollout (TR), and analytical (AM) manifold representations. The data is aligned such that contact occurs at $t = 0$. The performance of the CPF is drawn as a dashed lined.

4.1.3.4 Spatio-Temporal Sensor Resolution (H2–H3)

We evaluated the effect of sensor resolution on estimation accuracy by varying the resolution of binary contact sensors. In all cases, the sensors are distributed uniformly over the front surface of the hand. Figure 4.6b shows the relative performance of the CPF and MPF for three different resolutions averaged over 95 trials. As expected, the CPF performs worse as the spatial sensor resolution increases. In contrast, the MPF performs better. This confirms hypothesis H2.

Additionally, we measured the performance of the filters as we varied the distance traveled between sensor updates from 5 mm to 4 cm. Since the hand was moving at a constant velocity, this corresponds to changing the sensor's update frequency. Figure 4.6c shows the performance of the CPF and MPF averaged over 95 trials. As expected, the CPF performs worse as the update frequency increases. In contrast, the MPF performs better and confirms hypothesis H3.

4.1.3.5 Contact Manifold Representation (H4–H5)

We also compared the performance of the MPF using the RS, TR, and AM representations of the observable contact manifold. The RS representation consisted of 10,000 samples that were held constant throughout all of the experiments. The TR representation generated a different set 10,000 samples for each experiment by collecting five samples each from 2000 trajectory rollouts using the same physics model as used during execution. The AM was built using the parameters described in section 4.1.3.1.

Figure 4.7 shows the performance of the three representations averaged over 900 trials. The MPF outperformed the CPF with all three representations. As expected, the data supports hypothesis

H5: MPF-AM and MPF-TR both outperform MPF-RS. This occurs because the RS representation attempts to cover the entire surface S_c with a coarse set of samples. In contrast, the TR representation focuses the same number of samples on the smaller region of S_c that is encountered during execution.

Surprisingly, however, hypothesis H4 was not supported by the data: MPF-AM did not achieve lower error than MPF-TR representation. This is partially explained by same reasoning as above: the TR representation was able to densely cover the reachable subset of S_c at a resolution indistinguishable from that of the AM representation. Additionally, we know that every sample drawn from the TR representation must be reachable from the initial belief $b(s_0)$. This means that MPF-TR does not waste samples from the dual proposal distribution in regions of S_c that are known to be unreachable.

4.1.3.6 Sampling Failures

Our intuition is that the relatively poor performance of the MPF-RS is a result of it frequently failing to sample from the dual proposal distribution. Sampling from the dual proposal distribution fails when all particles sampled from $\bar{q}(s_t|S_c)$ have low probability $p(o_t|s_t, a_t)$ of generating o_t. In the case of binary contact sensors, a sampling failure typically occurs when several sensors are simultaneously active at runtime that were never observed to be simultaneously active while building the contact manifold representation.

Figure 4.8 shows the rate of sampling failures for the MPF-AM, MPF-RS, and MPF-TR computed over 900 trials. We formally define a sampling failure as an update where $p(o_t|s_t, a_t) < 0.1$ for all $s_t \in \tilde{S}_c$. Since there is a 10% chance of receiving an erroneous observation during contact, this corresponds to $\tilde{S}_c$ containing no states that are consistent with o_t. Under this metric, the TR and AM representations fail to sample from the dual proposal distribution for $<$ 30% of updates. Conversely, the RS representation fails to sample $>$ 70% of updates. When sampling fails the MPF behaves identically to the CPF and suffers from the same problem of particle starvation. As a result, MPF-RS performs poorly compared to MPF-AM and MPF-TR.

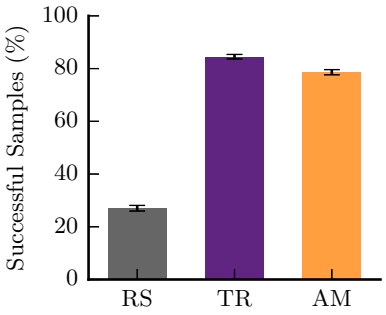


Figure 4.8: Percent of the time that the MPF succeeded at sampling from the dual proposal distribution during contact.

4.1.3.7 Mixing Rate

In addition to the manifold representation selected, the mixing rate parameter ϕ also has a strong impact on the performance of the MPF. We repeated the experiments described in section 4.1.3.3 while varying the MPF-AM's mixing rate over the set $\phi = \{0.025, 0.05, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0\}$. Note that $\phi = 1$ corresponds to the pure MPF. Figure 4.9 shows the post-contact performance of the MPF averaged

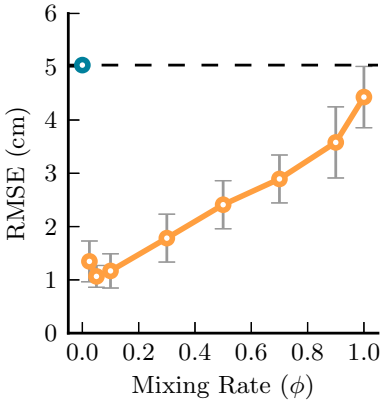


Figure 4.9: Performance of the MPF-AM as a function of the mixing rate $0 \leq \phi \leq 1$.

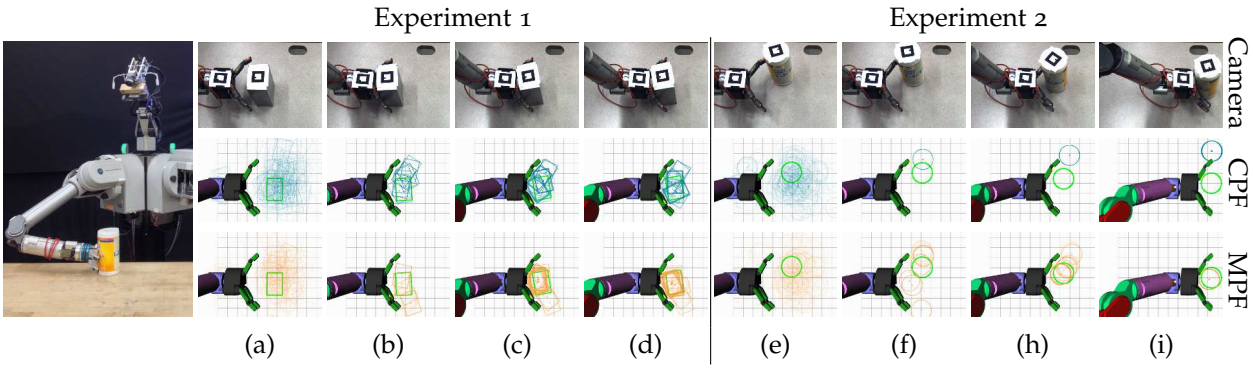


Figure 4.10: Andy pushing a box (a)–(d) and cylinder (e)–(i) across the table. The top row shows a video of the experiment from an overhead camera. The bottom two rows show the belief state estimated by the CPF (middle, dark blue) and MPF (bottom, light orange) as a cloud of particles. Ground truth is shown as a thick green outline.

over 150 trials and plotted as a function of ϕ. The performance of the CPF ($\phi = 0$) is plotted as a horizontal dotted line.

As expected, the MPF outperforms the CPF for all $\phi > 0$. Surprisingly, however, the optimal value of ϕ falls into the lowest range of ϕ values $0.025 \leq \phi \leq 0.1$ that we tested. Increasing ϕ out of this range leads to a predictable, linear increase in error. This occurs for two reasons. First, the dual proposal distribution performs poorly when there is observation noise [Thrun et al., 2000a]. Second, the MPF samples from an approximation of the dual proposal distribution that has higher variance than the true posterior belief. Reducing the mixing rate decreases the rate at which this variance grows.

4.1.4 *Real-Robot Experiments*

We evaluated the CPF and MPF on Andy [Bagnell et al., 2012], which used a Barrett WAM arm [Salisbury et al., 1988] equipped with the i-HY [Odhner et al., 2013] end-effector to push an object across a table. The i-HY's palm (48 tactels), interior of the proximal links (12 tactels each), interior of the distal links (6 tactels each), and fingertips (2 tactels each) were equipped with the array of tactile sensors [Tenzer et al., 2014] shown in fig. 4.11. The tactels were grouped into 39 vertical stripes to compensate for dead tactels and to simplify the observation model.

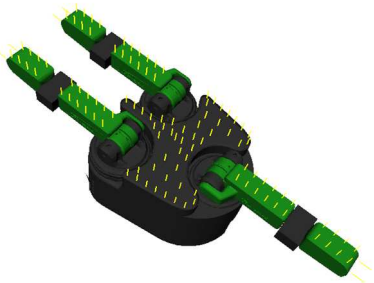


Figure 4.11: Tactile sensors (yellow lines) on the i-HY hand.

Figure 4.10 shows two representative runs of the state estimator on Andy. The ground-truth pose of the object was tracked by an overhead camera using a visual fiducial. Both filters were run with 250 particles, with $n = 250$ for the CPF and $n = 225$, $d = 25$, $\phi = 0.1$ for the MPF, and were updated after each 5 mm of end-effector motion. With the speed of the arm, this corresponded to an update rate of approximately 5–15 Hz.

In Experiment 1, Andy pushed a metal box that made initial contact with the right proximal link (b) and rolled into the palm (c). The CPF did not have any particles in the small observation space and,

thus, failed to track the box as it rolled into the palm (d). The MPF successfully tracked the box by sampling particles that agree with the observation. Note that the MPF was able to exploit the observation of simultaneous contact on the palm and distal link to correctly infer the orientation of the box.

In Experiment 2, Andy pushed a cylindrical container that made initial contact with its left fingertip (e). The cylinder rolled down the distal (f) and proximal (h) links to finally settle in the palm (i). Both the CPF and MPF made use of the initial contact observation to localize the container near the robot's left fingertip. However, the CPF's few remaining particles incorrectly rolled off of the fingertip and outside the hand. The MPF avoided particle starvation near the true state and was able to successfully track the container for the duration of contact.

4.2 *Pre- and Post-Contact Policy Decomposition*

The manifold particle filter allows us to filter the belief state $b(s_t)$, but does tell us which actions to take to achieve the goal. Formally, our goal is to find a policy $\pi : \Delta \to A$ that maximizes the value $V^\pi[b(s_0)]$ of the robot's initial belief state $b(s_0)$. Finding the optimal policy $\pi^* = \arg\max_\pi V^\pi[b(s_0)]$ requires solving a POMDP continuous state and action spaces, a large observation space, and discontinuous belief dynamics. This is intractable for most problems.

We continue to use s to refer to the object pose relative to the hand x_{rel} and o_t to refer to a contact sensor observation o_c in this section.

Our key insight is that π^* naturally decomposes (section 4.2.1) into two stages: (1) an open-loop *pre-contact trajectory* followed by (2) a closed-loop *post-contact policy* that uses sensor feedback to achieve success. This decomposition mirrors the dichotomy between gross (pre-contact) and fine (post-contact) motion planning found in early manipulation research [Hwang and Ahuja, 1992].

In this section, we describe how to exploit this structure to efficiently find a near-optimal policy. First, as an offline pre-computation step, we find a post-contact policy (section 4.2.3) by discretizing the state space and using a point-based POMDP solver [Kurniawati et al., 2008]. This is possible because we only need to plan for the set of beliefs whose support lies entirely on the contact manifold. Then, when presented with a new scene, we perform an efficient online search (section 4.2.4) to plan a pre-contact trajectory to make contact.

4.2.1 *Policy Decomposition*

Our key insight is that the discriminative nature of contact sensors naturally splits the optimal policy π^* into pre- and post-contact stages (insight 2). Before observing contact, the robot executes an

open-loop *pre-contact trajectory* $\xi \in A \times A \times \dots$ and receives a series of no-contact observations $o_1 = \dots = o_{t-1} = o_{\text{nc}}$. Once contact is observed $o_t \in O_{\text{c}}$, the closed-loop *post-contact policy* π_{c} uses feedback from the hand's contact sensors to achieve the goal.

Decomposing the policy into pre- and post-contact stages is equivalent to splitting the value function

$$V(b) = \max_{a \in A} \Bigg[R(b,a) + \gamma \int_{\Delta} T(b,a,b') \Bigg(\underbrace{\Omega(o_{\text{nc}}, b', a) V(b')}_{\text{pre-contact}} + \underbrace{\sum_{o \in O_{\text{c}}} \Omega(o, b', a) V^{\text{c}}(b')}_{\text{post-contact}} \Bigg) db' \Bigg] \tag{4.5}$$

into two separate terms that depend on the current observation and the value function V^{c} of the post-contact policy π_{c}. We only need to consider the current observation—instead of the full belief b—because contact sensors accurately discriminate between contact ($o \in O_{\text{c}}$) and no-contact ($o = o_{\text{nc}}$).

The pre-contact term is active only for $o = o_{\text{nc}}$ and includes the reward earned from executing the remainder of ξ. Conversely, the post-contact term is active for the remaining observations $o \in O_{\text{c}}$ and includes all reward $V^{\text{c}}(b)$ that would be earned by π if the robot were to observe contact in b and immediately switch to executing the post-contact policy.

We compute the post-contact policy π_{c}—and corresponding value function V^{c}—once per hand-object pair using a point-based method in an offline pre-computation step (section 4.2.3). Our intuition, as shown in fig. 4.14, is that exhaustively computing this policy is tractable because the set of post-contact beliefs $\Delta_o \subseteq \Delta$ is relatively small. Then, when given a problem instance, we solve for the pre-contact trajectory ξ that is optimal with respect to π_{c} using an online search. As shown in fig. 4.12, this is equivalent to truncating an online POMDP search once contact has occurred and using V^{c} to evaluate the value of the truncated subtrees.

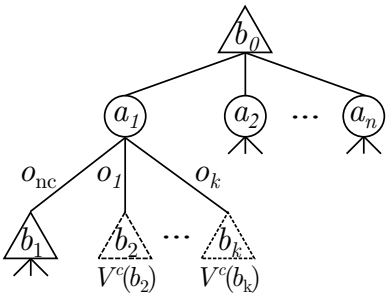


Figure 4.12: Online POMDP solvers must branch over both A and O. The pre-contact search only branches over A by evaluating all post-contact belief states with the post-contact value function V^{c}.

4.2.2 *Suboptimality Bound*

Factoring the policy into pre-contact and post-contact components has a bounded impact on the performance of the overall policy. To prove this, we assume that the pre- and post-contact stages share identical discrete state, action, and observation spaces and consider a search of depth T. Under these circumstances, error can come from two sources: (1) truncating the pre-contact search and (2) using a sub-optimal post-contact value function.

We derive an explicit error bound on $\eta = ||V - V^*||_\infty$ by recursively expanding the Bellman equation for the for T-horizon policy V_T in terms of the value function V_{T-1} of the $(T-1)$-horizon policy:

$$\begin{aligned} ||V_T - V^*||_\infty &\leq \gamma ||V_{T-1} - V^*||_\infty + \gamma P_{\max} ||V^{\mathrm{c}} - V^*||_\infty \\ &\leq \gamma^T ||V_0 - V^*||_\infty + \sum_{t=1}^{T} \gamma^t P_{\max} ||V^{\mathrm{c}} - V^*||_\infty \\ \eta &\leq \gamma^T \eta_{\mathrm{nc}} + \tfrac{\gamma(1-\gamma^T)}{1-\gamma} P_{\max} \eta_{\mathrm{c}} \end{aligned} \tag{4.6}$$

First, we distribute $||\cdot||_\infty$ using the triangle inequality and bound the maximum single-step probability of contact with $0 \leq P_{\max} \leq 1$. Next, we recursively expand V_T in terms of V_{T-1} until we reach the evaluation function V_0 used to approximate the value V_{T+1} of the truncated sub-tree. Finally, we evaluate the geometric series and express the result in terms of the sub-optimality of our evaluation function $\eta_{\mathrm{nc}} = ||V_0 - V^*||_\infty$ and post-contact policy $\eta_{\mathrm{c}} = ||V^{\mathrm{c}} - V^*||_\infty$. In the worst case we can bound $\eta_{\mathrm{nc}} \leq -R_{\min}/(1-\gamma)$ by setting $V_0 = 0$ since the reward function is bounded by $R_{\min} \leq R(s,a) \leq 0$.

Often V_0 is formulated as the value function of a *default policy* π_{default} in online POMDP planning literature [Silver and Veness, 2010, Somani et al., 2013].

As expected, eq. (4.6) shows that $\eta \to 0$ as $\eta_{\mathrm{c}}, \eta_{\mathrm{nc}} \to 0$. This is the sameresult as in traditional online POMDP search algorithms [Ross et al., 2008]. However, error does not approach zero as $T \to \infty$ because the full policy can never outperform a sub-optimal post-contact policy π_{c}.

4.2.3 *Post-Contact Policy*

Suppose the robot is in belief state b while executing ξ, takes action a, receives contact observation $o \in O_{\mathrm{c}}$, and transitions to the posterior belief state b'. At this point—due to the discriminative nature of contact sensors—we know that the object is in non-penetrating contact with one or more contact sensors. Formally, we know that the true state $s \in S$ lies on the *observable contact manifold* $S_{\mathrm{c}} \subseteq S_{\mathrm{c}}$, the set of all object poses that are in contact with one or more sensors (insight 1).

Figure 4.13 shows S_{c} for a hand manipulating a rectangular box. The contact manifold is a two-dimension structure embedded in $SE(2)$ where the vertical axis represents the orientation of the object relative to the hand. Each color in on the manifold represents a region of S_{c} that is in contact with a unique sensor. White regions of manifold indicate contact between the object and multiple sensors.

Since the state is known to lie on S_{c}, we additionally know that the belief state b' is in *post-contact belief space* $\Delta_o = \{b \in \Delta : b(s) = 0 \; \forall \, s \notin S_{\mathrm{c}}\}$ with support constrained to S_{c} (fig. 4.14a). Furthermore, many belief states $\mathcal{R}(\Delta_o)$ reachable from Δ_o are constrained to S_{c} because the state evolves on S_{c} during contact. As a result, the post-contact

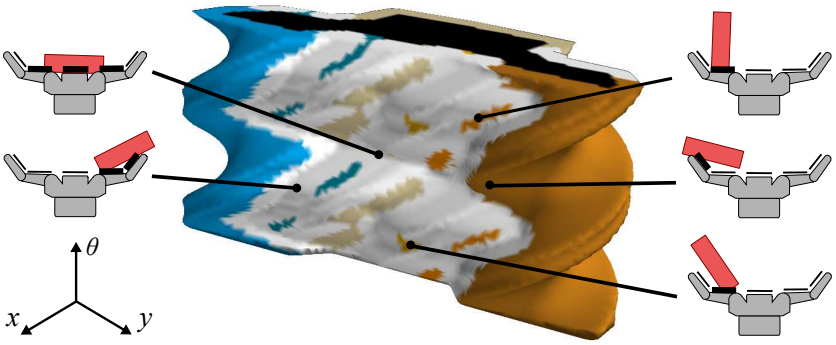


Figure 4.13: The observable contact manifold S_c for a hand manipulating a rectangular box. Each color indicates that the object is in contact with a particular sensor. White indicates contact with multiple sensors.

POMDP is particularly well suited to being solved by a point-based method [Lee et al., 2007].

Point-based methods, first introduced by Pineau et al. [2003] are a class of offline POMDP solvers that break the *curse of history* by performing backups at a discrete set of *belief points* $B \subseteq \mathcal{R}(b_0)$. These methods perform well when the *reachable belief space* $\mathcal{R}(b_0) \subseteq \Delta$, the set of beliefs that are reachable from the initial belief b_0 given any sequence of actions and observations, is small [Pineau et al., 2003].

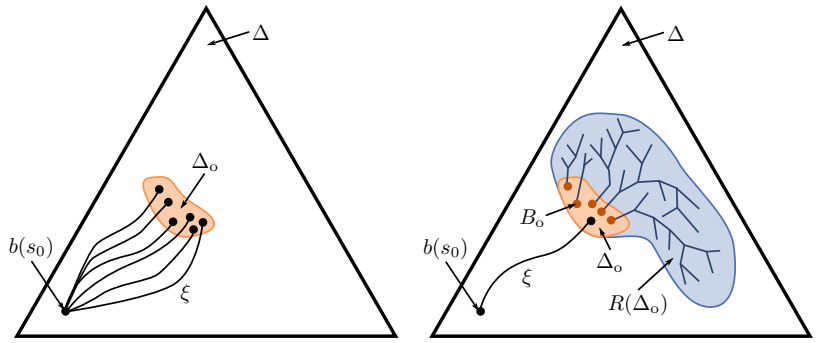
4.2.3.1 Initial Belief Points

Traditionally, point-based POMDP solvers assume that the initial belief state $b_0 \in \Delta$ is known. In our application, we only know that $b_0 \in \Delta_o$ occurred after the pre-contact trajectory terminated. We cannot seed the point-based solver with the full set of post-contact belief states because Δ_o is uncountably infinite. Instead, we initialize the solver with with a discrete set of samples $B_o \subseteq \Delta_o$ similar to the initial beliefs that we expect to see during execution (fig. 4.14b). Since we expect a contact sensor to produce a uni-modal belief state, we initialize B_o with Gaussian distributions with means distributed over S_{trust}. This initial set of belief points can be refined over time by adding the post-contact beliefs observed during execution to B_o.

In the simplest case, we can independently solve for a separate policy for each belief point $b_i \in B_o$. The post-contact policy over B_o consists of the union of the α-vectors for each of these policies. Since each α-vector is a global lower bound on the true value function, the union of the α-vectors is better approximation of the value function than any one of the policies in isolation. This approach, however, ignores the fact that information can be shared between policies; i.e. it is generally the case that $\mathcal{R}(b_i) \cap \mathcal{R}(b_j) \neq \emptyset$ for $i \neq j$.

We leverage this structure by converting our POMDP with set of initial belief states B into an augmented POMDP with one initial belief. By doing so, we allow the POMDP solver to exploit its belief

See Koval et al. [2015c] for more information about the reduction of the set of initial belief points Δ_o belief points to an augmented POMDP with one initial belief point.



(a) Pre-Contact Trajectory (b) Post-Contact Policy

Figure 4.14: We decompose the policy π into a (a) pre-contact trajectory ξ and a (b) post-contact policy π_c. Pre- and post-contact are coupled by the small set of post-contact belief states $\Delta_o \subseteq \Delta$.

point expansion heuristics to efficiently find a policy over all of Δ_o.

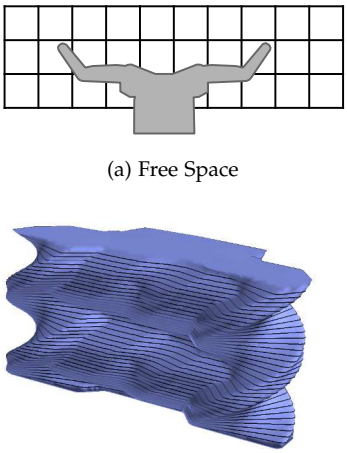
4.2.3.2 State Space Discretization

Most point-based solvers—with few exceptions [Porta et al., 2006, Brunskill et al., 2008, Bai et al., 2011]—are restricted to discrete state, action, and observation spaces. Manipulation occurs in continuous state and action spaces, so we must discretize S and A to apply these techniques. We propose alternatives to a priori discretization of the state space in section 5.1.

Ideally, we would only discretize the regions of S that comprise the support of the optimally-reachable belief space $\mathcal{R}^*(\Delta_o)$. Unfortunately, finding the optimally-reachable belief space is just as hard as solving the full POMDP [Kurniawati et al., 2008]. Instead, we define a *trust region* $S_{\text{trust}} \subseteq S$ that we believe to over-estimate the support of $\mathcal{R}^*(\Delta_o)$ and solve the post-contact POMDP over this region.

There is a trade-off in choosing the size of the trust region: making S_{trust} too small may disallow the optimal policy, while making S_{trust} too large will make it intractable to solve the resulting POMDP. In the case of quasistatic manipulation [Mason, 1986], we believe S_{trust} to be relatively small because the optimal policy will not allow the object to stray too far from the hand. Note, however, that requiring S_{trust} to be small disallows policies that require large global motions; e.g. performing an orthogonal push-grasp once the object has been localized along one axis.

We compute the discrete transition, observation, and reward functions over S_{trust} by taking an expectation over the continuous models under the assumption that there is a uniform distribution over the underlying continuous states. In practice, we approximate the expectation through Monte Carlo rollouts. It is important to preserve the structure of the underlying continuous state space when discretizing S_{trust}. Otherwise, the discrete model will not obey the Markov property when executed on the real robot: the output of the transition,



(a) Free Space

(b) Contact Manifold

Figure 4.15: Explicit discretization of (a) free space and (b) the contact manifold.

Assuming there is a uniform distribution over the discrete state is motivated by the *principle of maximum entropy* [Jaynes, 1957]. We discuss this further in section 5.1.

observation, or reward function will depend on on the underlying continuous state, rather than only the discrete state.

Uniformly discretizing S_{trust} is a poor choice because contact is inherently discontinuous: two states in S may be arbitrarily close together, but behave vastly differently depending upon whether the object is in contact with the hand (challenge 1). The robot can only push an object while in contact with it. Similarly, contact sensors only activate when the hand is in contact with the object.

We explicitly model the discontinuity of contact by composing the trust region S_{trust} from two components: (1) a uniform discretization of free space $S_{\text{nc}} = S_{\text{trust}} \setminus S_{\text{c}}$ (fig. 4.15a) and (2) an explicit discretization of the contact manifold S_{c} (fig. 4.15b). We represent the contact manifold using the analytic representation described in section 4.1.2.2.

4.2.4 *Pre-Contact Trajectory*

The belief dynamics are a deterministic function of the action given a fixed sequence of "no contact" observations. As a result, we can find the optimal trajectory ξ (fig. 4.14b) by running an optimal graph search algorithm, such as A* [Hart et al., 1968], in an augmented belief space by recursively expanding V in eq. (4.5) to

$$V(b) = \max_{\xi} \left[\sum_{t=0}^{\infty} \gamma^t \left(\prod_{i=0}^{t} \Omega(o_{\text{nc}}, b_{i+1}, a_i) \right) \Big(R(b_{t+1}, a_t) + \sum_{o \in O_c} \Omega(o, b_{t+1}, a_i) V^{\text{c}}(b_{t+1}) \Big) \right]. \tag{4.7}$$

Each term in the summation corresponds to taking a single action in ξ. The product over $t = 0, \ldots, t$ is equal to the probability of reaching time t without having observed contact.

4.2.4.1 Graph Construction

Define a directed graph $G = (V, E, c)$ where each node $x = (b, p_{\text{nc}}, t) \in V$ consists of a belief state b, the probability p_{nc} of having not yet observed contact, and the time t. An edge $(x, x') \in E$ corresponds to taking an action a in belief state b and transitioning to belief state b'.

The cost of an edge $(x, x') \in E$ from $x = (b, p_{\text{nc}}, t)$ to $x' = (b', p'_{\text{nc}}, t')$ is

$$c(x, x') = -\gamma^t p_{\text{nc}} \left(R(b', a) + \gamma \sum_{o \in O_c} \Omega(o, b', a) V^{\text{c}}(b') \right),$$

precisely one term in the above summation. The no-contact probability evolves as $t' = t + 1$ and $p'_{\text{nc}} = p_{\text{nc}} \Omega(o_{\text{nc}}, b', a)$ because the Markov

property guarantees $o_t \perp o_{t-1} | s_t$. When $p_{\text{nc}} = 0$ the cost of executing ξ is $-V(b_0)$. Therefore, finding the minimum-cost ξ is equivalent to finding the optimal value function $V(b_0)$.

Intuitively, the cost of an edge consists of two parts: (1) the immediate reward $R(b', a)$ from taking action a in belief state b and (2) the expected reward $\sum_{o \in O_c} \Omega(o, b', a) V^c(b')$ obtained by executing π_c starting from b'. The minimum-cost path trades off between making contact quickly (to reduce p_{nc}) and passing through beliefs that have high value under π_c.

4.2.4.2 Heuristic Function

The purpose of a heuristic function is to improve the efficiency of the search by guiding it in promising directions. Heuristic-based search algorithms require that the heuristic function is *admissible* by underestimating the cost to goal and *consistent* [Pearl, 1984]. Heuristic functions are typically designed by finding the optimal solution to a relaxed form of the original problem.

Since the true cost to the goal from a particular belief is the negated value function, we compute a lower bound on the cost-to-come by computing an upper bound on the value function. We intentionally choose a weak, computationally inexpensive heuristic function since the pre-contact search primarily explores the simple, no-contact regions of belief space.

We do this by solving for the value function of the *MDP approximation* [Ross et al., 2008] of our problem. The value function $V^{\text{MDP}}(s)$ of the optimal policy for the MDP (S, A, T, R) is an upper bound

$$V^*(b) \leq V^{\text{MDP}}(b) = \int_S V^{\text{MDP}}(s) b(s) ds$$

on the POMDP value function $V(b)$.

Next, we upper-bound the MDP value function V^{MDP} with a deterministic search in the underlying state-action space by ignoring stochasticity in the transition function. Finally, we compute an upper-bound on the value of the graph search by lower-bounding the cost of the optimal path with a straight-line motion of the hand that is allowed to pass through obstacles.

After making these assumptions, the MDP approximation of the value function is

$$V^{\text{MDP}}(s) \leq \sum_{t=0}^{t_{\min}} \gamma^t R_{\max}$$

where $R_{\max} = \max_{a \in A, s \in S} R(s, a)$ is the maximum reward and $t_{\min}$ is the minimum number of steps required to make contact with the

object. We can compute a lower bound on bound on $t_{\min}$ as

$$t_{\min} = \left\lfloor \frac{\min_{s' \in G} \text{dist}(s, s')}{d_{\max}} \right\rfloor$$

where $\text{dist}(s, s')$ is the straight line distance between two positions of the states, and $d_{\max}$ is the maximum displacement of all actions. Note that we cannot simply use $\text{dist}_{s' \in G}(s, s')$ as the heuristic because it omits the discount factor γ.

This is an upper bound on V^{MDP} because we are over-estimating reward and under-estimating the time required to achieve the reward in an environment with $R(s, a) \leq 0$. Therefore, from the definition of the MDP approximation [Ross et al., 2008], we know that

$$h(x) = \gamma^t p_{\text{nc}} \int_S V^{\text{MDP}}(s) b(s) ds.$$

is an admissible heuristic for state $x = (b, t, p_{\text{nc}})$.

4.2.4.3 Search Algorithm

We employ weighted A*, a variant of A*, to search the graph for an optimal path to the goal [Pohl, 1977]. Weighted A* operates identically to A* but sorts the nodes in the frontier with the priority function

$$f(v) = g(v) + \epsilon_w h(v)$$

where $g(v)$ is the cost-to-come, $h(v)$ is the heuristic function, and ϵ_w is the heuristic inflation value.

For $\epsilon_w > 1$, weighted A* is no longer guaranteed to return the optimal path, but the cost of the solution returned is guaranteed to be no more than ϵ_w times the solution cost of the true optimal path [Pohl, 1977]. Weighted A* has no bounds on number of expansions, but—in practice—expands fewer nodes that A*. This is beneficial when, such as in our case, it is computationally expensive to generate successor nodes.

4.2.5 Simulation Experiments

We evaluated the performance of the policies produced by our algorithm in simulation experiments. First, we evaluate the performance of the post-contact policies produced for our discretization of the contact manipulation POMDP against a baseline QMDP policy [Littman et al., 1995]. The QMDP policy assumes that uncertainty disappears after one time step and, thus, does not plan to take multi-step information gathering actions.

4.2.5.1 Hypotheses

Specifically considering the post-contact policy, we hypothesize:

H1. *The POMDP post-contact policy will achieve higher value than the baseline QMDP policy.*

H2. *The POMDP post-contact policy will achieve success with higher probability than the baseline QMDP policy.*

The key difference between these two hypotheses is that H1 tests how well the POMDP policy performs on the discrete state space used for planning. Conversely, H2 tests how well performance on the discrete state space translates to achieving the task in the true, continuous state space. This confirms that our discretization faithfully matches the underlying continuous belief dynamics.

However, in practice, we are interested in the performance of the full pre- and post-contact policy. We hypothesize:

H3. *The pre-contact trajectory will not significantly effect the success rate of either post-contact policy.*

H4. *The full POMDP policy will outperform the full QMDP policy.*

Our performance bound suggests that decomposing the problem into pre- and post-contact stages should not, as H3 states, significantly harm performance. As a result, H4 states that we do not expect the relative performance of the POMDP and QMDP policies to change.

Finally, we must consider the effect of sensor coverage on the performance of these policies. Ideally, increasing sensor coverage should reduce the burden on the planner to perform information-gathering actions. Therefore, we hypothesize:

H5. *Both policies will improve when provided with better sensor coverage.*

H6. *The QMDP policy will improve more than the POMDP policy.*

Improved sensor coverage always reduces uncertainty and, thus, should improve the performance of both policies. However, since the QMDP baseline is optimal when state is fully observed, we expect the improvement to be larger for the QMDP than for the POMDP policy.

4.2.5.2 Experimental Design

We simulated our algorithm in the same scenario we used to evaluate the MPF in section 4.1.3. We use a set of $|A| = 5$ purely translational actions with length $||a_i|| \approx 2$ cm for planning. This set includes forward, left, right, and diagonal actions. We assigned the diagonal actions length $2\sqrt{2}$ cm to force all five actions to align with our discretization of the post-contact state space.

Actions that are not aligned with the discretization can lead to discretization artifacts that affect the performance of the post-contact policy. We propose to address this limitation in section 5.1.

We use a particle filter (section 4.1) to perform belief updates during execution of the pre-contact policy. Once we transition to the post-contact policy, e.g. by receiving a contact observation, we immediately convert the robot's continuous belief state into a discrete belief state over the post-contact state space. From this point forward, the robot tracks its discrete belief state using a discrete Bayes filter [Thrun et al., 2005]. This ensures that the state estimator used during execution matches the belief dynamics used during planning.

Our use of the discrete Bayes filter to track the belief state during execution of the post-contact policy differs from the experiments presented in earlier versions of this work [Koval et al., 2014]. In those experiments, we used the manifold particle filter to track state during execution of the full policy. As a result, the results that we present in this paper slightly differ from those presented in prior work.

We selected a trust region S_{trust} of size 15 cm $\times$ 50 cm around the hand. We discretized the no-contact S_{nc} portion of S_{trust} at a coarse 2 cm $\times$ 2 cm $\times$ 12° resolution and the contact S_{c} portion of S_{trust} at a higher 1 cm $\times$ 12° resolution. The set S_{c} was represented by computing an analytic representation of the contact manifold (section 4.1.2.2). The discrete state space contained $|S| = 4239$ states split between $|S_{\text{nc}}| = 2625$, $|S_{\text{c}}| = 1613$, and one unknown state.

4.2.5.3 Dependent Measures

We compared the POMDP policy generated by our algorithm to a policy that is greedy with respect to the QMDP value function. The QMDP value function is an upper-bound on the POMDP value function constructed by solving the underlying MDP [Littman et al., 1995]. Intuitively, the QMDP value function assumes that all uncertainty is resolved after taking each action. This allows a QMDP policy to advantage of observations during execution, but not execute multi-step information-gathering actions to reduce uncertainty.

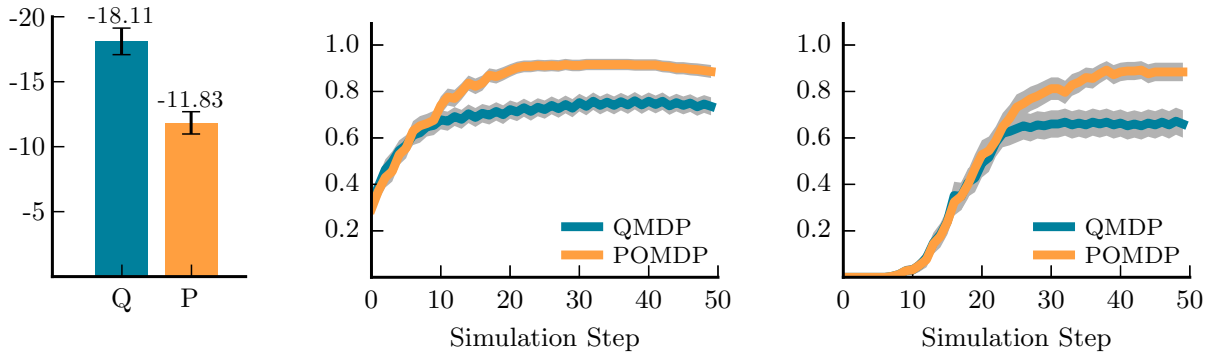
Ideally we would also compare against a full policy computed by SARSOP. Unfortunately, we were not able to get implementation of SARSOP provided by the APPL toolkit [Kurniawati et al., 2008] to successfully load the full discrete POMDP model. We discuss this further in section 5.1.3.

We evaluate the performance of the two algorithms' post-contact policy on the discrete state space by measuring the expected value of the policy over a horizon of length 50. We approximate the expected value by performing a large number of rollouts and averaging the result. We also measure the *success rate* of the policy by computing the probability $\Pr(s \in G)$ of the object while simulating its motion using the continuous system dynamics.

4.2.5.4 Post-Contact Policy (H1–H2)

We solved for the post-contact QMDP policy by running MDP value iteration on the discrete POMDP model until the α-vectors converged within 10^{-6}. This process took 1729 backups over 8.36 s seconds (4.84 ms per backup) for the box. The resulting policy consisted of five α-vectors and took 7.64 ms to evaluate on a discrete belief state.

We repeated this procedure to generate the post-contact POMDP policy by running a point-based solver on $|B| = 15$ initial belief points, each assigned equal weight. Each belief point $b_i \in B$ was a Gaussian distribution $b_i \sim \mathcal{N}(\mu_i, \Sigma_i)$ with mean $\mu_i = [x_i, y_i, \theta_i]^T$ and



(a) Discrete Value

(b) Post-Contact Success Rate

(c) Full Policy Success Rate

Figure 4.16: Performance of the post-contact policy. (a) Expected value, at depth 50, of the QMDP (Q) and POMDP (P) post-contact policies evaluated on the discrete system dynamics. Reward is always negative, so less-negative values are better. Success rate of the (b) post-contact and (c) full policy evaluated on the continuous system dynamics. The error bars and gray shaded region denote a 95% confidence interval.

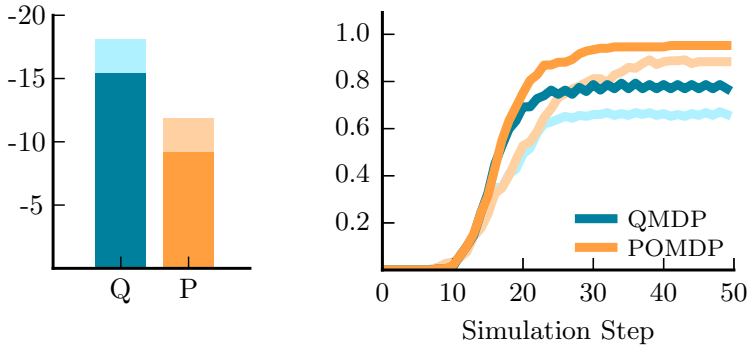
covariance matrix $\Sigma_i^{1/2} = \text{diag}[5\text{ cm}, 5\text{ cm}, 15^\circ]$. The mean of this distribution was a fixed distance $x = 0.2$ m in front of the hand with a lateral offset selected uniformly range $-10\text{ cm} \leq y \leq 10\text{ cm}$ and an orientation selected uniformly $0 \leq \theta < 360^\circ$.

We solved the resulting discrete POMDP using the SARSOP implementation provided by APPL Tookit [Kurniawati et al., 2008]. We ran SARSOP for 5 minutes, during which it produced a policy consisted of several thousand α-vectors. We intentionally ran SARSOP until convergence, just as we did for QMDP, to eliminate this as a variable in our analysis: it may be possible to terminate planning much earlier—resulting in a more compact policy—with negligible impact on runtime performance of the policy.

We evaluated the quality of the QMDP and POMDP post-contact policies on the discrete system dynamics performing 1000 rollouts on initial belief states drawn from B for each policy. Figure 4.16a shows that the POMDP policy significantly outperformed the QMDP policy. This result confirms hypothesis H1: it is advantageous to formulate the contact manipulation problem as a POMDP.

Next, we executed 500 rollouts of the policy using the continuous transition model to simulate the motion of the object and the continuous observation model to generate observations. The initial belief states for each experiment was sampled uniformly at random from the range described above. These belief states were not contained in B. At each timestep, we recorded whether the simulated object was in the goal region G; i.e. achieved success.

Figure 4.16b shows that the higher-quality discrete POMDP policy translates to a high-quality policy in the continuous system dynamics. The POMDP policy successfully achieves the goal both more quickly and with higher probability than the QMDP policy. This confirms hypothesis H2: our discrete POMDP and choice of reward function succeed in driving the object into G.



(a) Discrete Value

(b) Full Policy Success Rate

Figure 4.17: Effect on (a) discrete value and (b) success rate of increasing sensor coverage from only the fingertips (faded colors) to the entire surface of the hand (solid colors). Again, note that less-negative values are better. Error bars are omitted to reduce visual clutter.

4.2.5.5 Pre-Contact Trajectory (H3–H4)

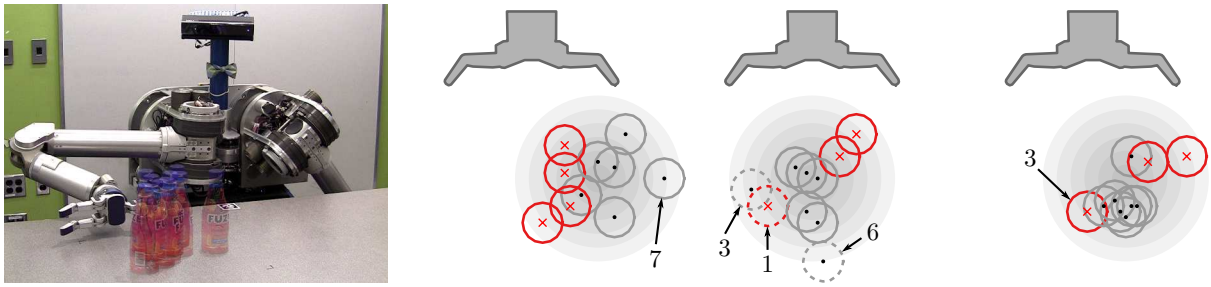
To find the pre-contact trajectory ξ, we ran a separate weighted A* search for each post-contact policy with a heuristic inflation factor of $\epsilon_w = 2$. The search terminated once a node was expanded that satisfied one of the following criteria: (1) ξ achieved contact with 100% probability, (2) 85% of the remaining belief lied in S_{trust}, or (3) the search timed out after 20 s.

We sampled 250 prior beliefs with $\Sigma^{1/2} = \text{diag}[5\text{ cm}^2, 5\text{ cm}^2, 15^\circ]$ variance and a mean located 0.5 m in front of and up to 0.5 m laterally offset from the center of the palm. Note that all of these beliefs lie significantly outside of the trust region and it is not possible to directly execute the post-contact policy.

Figure 4.16c shows the success rate of the robot executing the combined pre- and post-contact policy on both the disk and the box. As expected, each algorithm achieved a success rate that is comparable to the that of the underlying post-contact policy. This confirms hypothesis H3: the pre-contact trajectory faithfully extends the post-contact policy to longer horizons. Additionally, these results confirm hypothesis H4: the POMDP policy outperforms the QMDP policy when executed at the end of the pre-contact trajectory.

4.2.5.6 Sensor Coverage (H5–H6)

Finally, we analyzed the impact of sensor coverage on the performance of the QMDP and POMDP policies. We considered an improved observation model where each of the hand's $n = 7$ links served as a binary contact sensor. We analyzed the geometry the analytic representation of the contact manifold to compute that and ten (box) observations, of the $|O| = 2^7 = 128$ total, are geometrically feasible. This is a large increase in sensing capability over the three observations that were geometrically feasible with the fingertip sensors.



(a) Experimental Setup (b) Belief 1 (c) Belief 2 (d) Belief 3

Figure 4.18: (a) HERB in the initial configuration used to begin each trial. (b)–(d) Samples (circles) drawn from the three initial belief states (light gray iso-contours) used for evaluation. Samples are drawn with a black dot in the center if the QMDP policy succeeded and a red $\times$ if it failed. In (c) belief 2, the three samples on which SARSOP failed are drawn with a dotted line. Labels correspond to the trial numbers from table 4.1.

Figure 4.17a shows that increasing sensor coverage improves expected post-contact value. The improved performance of the post-contact policy directly translates to better performance of the full pre- and post-contact policies. Figure 4.17b shows that increasing sensor coverage significantly improves both the success rate and speed of both policies. Also, as expected, there is a larger improvement for the QMDP policy than the POMDP policy. This confirms H5 and H6.

4.2.6 *Real-Robot Experiments*

We executed the policies produced by the QMDP and SARSOP algorithms on HERB [Srinivasa et al., 2012], which is 7-DOF Barrett WAM arm [Salisbury et al., 1988] and the BarrettHand end-effector [Townsend, 2000]. We used HERB to push a bottle across the table (fig. 4.18a) using a similar experimental setup as in section 4.2.5.

We generated three Gaussian initial belief states with means $\mu_1 = [35\text{ cm}, 12\text{ cm}]$, $\mu_2 = [35\text{ cm}, 0\text{ cm}]$, and $\mu_3 = [35\text{ cm}, 6\text{ cm}]$ relative to HERB's hand. All three initial belief states had covariance $\Sigma^{1/2} = \text{diag}[5\text{ cm}, 5\text{ cm}]$ and the goal region was a 4 cm $\times$ 8 cm rectangle in front of HERB's palm. We sampled each Gaussian ten times and used the same thirty samples to evaluate both QMDP and SARSOP. The three initial belief states and thirty samples used for evaluation are shown in figs. 4.18b to 4.18d.

HERB used the BarrettHand's strain gauges to detect contact with the distal links of the fingers. The sensors were re-calibrated before each trial and were thresholded to produce a discriminative, binary contact/no-contact observation. Actions were executed using a Jacobian pseudo-inverse controller. Each trial terminated when the selected action was infeasible (e.g due to joint limits or collision) or the bottle reached the edge of the table. A trial is considered successful if the bottle was in the goal region at the moment of termination.

See Koval et al. [2015c] for a detailed description of the controller we used to execute the QMDP and SARSOP policies.

		1	2	3	4	5	6	7	8	9	10
Belief 1	QMDP	·	×	·	·	·	×	·*	·	×	×
	SARSOP	·	·	·	·	·	·	·	·	·	·
Belief 2	QMDP	×	·	·	×	·	·	·	×	·	·
	SARSOP	×†	·	×†	·	·	×‡	·	·	·	·
Belief 3	QMDP	·	·	×	·	×	·	×	·	·	·
	SARSOP	·	·	·	·	·	·	·	·	·	·

Table 4.1: Outcome of executing the QMDP and SARSOP policies on HERB. Each policy was executed on ten samples from three initial belief states. The symbol · denotes success and × failure to push the object into the goal region. The trial marked with ∗ succeeded because the object contacted a sensor during execution of the pre-contact trajectory, trials marked with † failed due to kinematics, and the trial marked with ‡ failed due to un-modelled errors in the observation model.

4.2.6.1 Experimental Results

Table 4.1 shows the outcome of all thirty trials executed on HERB [Srinivasa et al., 2012]. The position of the bottle used in each trial is shown in fig. 4.18. The SARSOP policy achieved success on 27/30 trials (90%). In contrast, the QMDP policy only achieved success on only 20/30 trials (67%).

Qualitatively, the pre-contact trajectories for both policies aligned the mean of the initial belief state with the center of the palm. Once the pre-contact trajectory was exhausted, the QMDP post-contact policy pushed straight, similar to the open-loop push grasp [Dogar and Srinivasa, 2010]. The SARSOP policy moved straight for several time steps, then executed a sideways action to localize the bottle by driving it into a contact sensor. Once the bottle was localized, the policy centered the bottle in the hand and began pushing pushing straight.

This is similar to the move-until-touch "guarded move" used in part assembly [Will and Grossman, 1975] that has been automatically synthesized by recent work in robotic touch-based localization [Petrovskaya and Khatib, 2011, Javdani et al., 2013, Hebert et al., 2013]. Figure 4.19-Bottom shows an example of this type of behavior.

All of the failures of the QMDP policy occurred because the bottle came to rest in a configuration that was: (1) outside of the goal region and (2) not in contact with a sensor. The QMDP policy chose the "move straight" action in this situation because moving to either side reduces the probability of the bottle being in the goal region. This highlights the key limitation of the QMDP approximation: the QMDP policy ignores uncertainty in future time steps and, thus, will not plan multi-step information gathering actions [Littman et al., 1995].

However, the QMDP policy does incorporate feedback from contact sensors during execution. This was critical in belief 1 trial 7 (marked with a ∗ in table 4.1, fig. 4.19-Top). In this trial, the QMDP policy achieved success despite the fact that the initial pose of the bottle began outside of the capture region of the open-loop push grasp (see fig. 4.18). The bottle came in contact with a sensor during execution of the pre-contact policy and, thus, was localized.

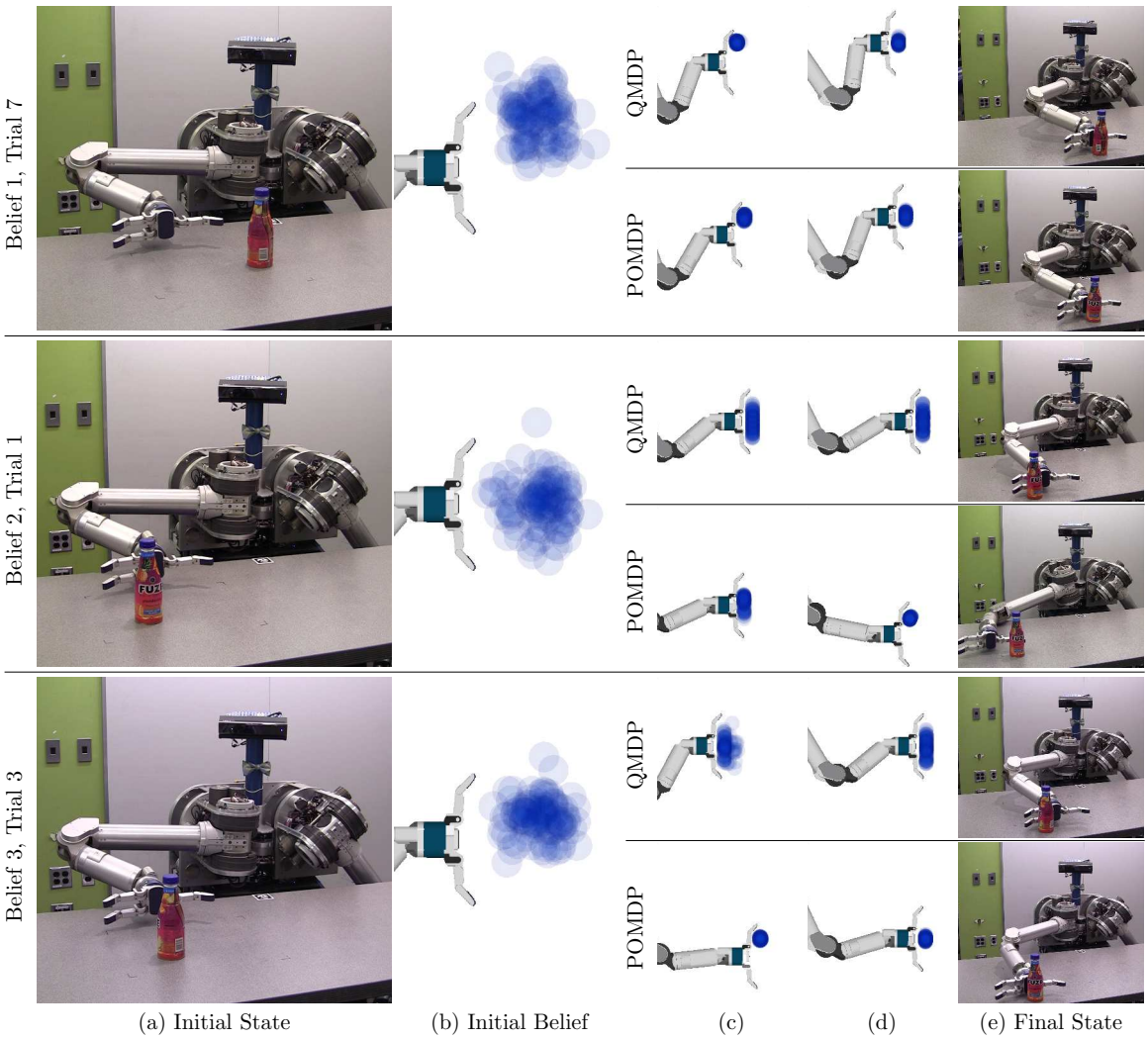


Figure 4.19: Three trials executed on HERB with the QMDP and POMDP policies. Columns (a) and (e) show, respectively, the initial and final pose of the object relative to the hand. Column (b) shows several samples drawn from the initial belief state. Columns (c), and (d) show the belief state over object pose at two points during execution. The trial numbers referenced in this figure correspond to those used in fig. 4.18 and table 4.1.

The SARSOP policy used information-gathering actions to localize the object regardless of its initial configuration. As a result, the SARSOP policy succeeded on 20/20 trials on beliefs 1 and 3. Surprisingly, the SARSOP policy failed on three trials from belief 2. Two trials (marked with † in table 4.1, fig. 4.19-Bottom shows one of these) occurred because HERB's elbow collided with the table while performing the "move-until-touch" action described above. In both cases, HERB could have avoided contact with the table by moving left (instead of right) to localize the bottle. We propose to plan with these types of kinematic constraints in section 5.2.

5
Proposed Work

We have taken a first step towards using real-time feedback from contact sensors to robustly manipulate objects under uncertainty [Koval et al., 2015c]. Our experimental results in section 4.2.5 show that contact sensing can be harnessed for real-time feedback during manipulation. However, we makes several assumptions that make it challenging to use our approach in practice.

First, we discretized the state space S to find a post-contact policy using an existing point-based POMDP solver. Discretization artifacts can, in some cases, significantly affect the performance of the policy during execution. We propose to find a near-optimal policy without relying on a coarse discretization of the state space (section 5.1).

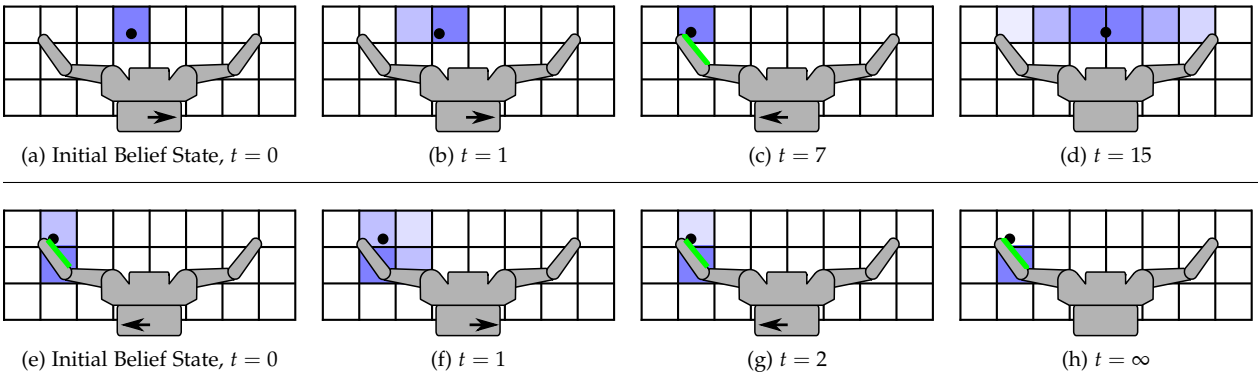
Second, we ignored kinematic constraints on the robot during planning (e.g joint limits and self collision). These constraints can cause the policy to fail during execution. We propose a simple technique to avoid these situations by enforcing kinematic constraints during planning (section 5.2).

Finally, we planned in the space of object poses relative to the hand. Since we often have imperfect proprioception (section 3.2), this restricts us to achieving goals that are expressed relative to the hand. We propose to consider proprioceptive uncertainty during planning to achieve goals that are expressed in the world frame (section 5.3).

5.1 Manipulation in Continuous Space

With few exception, manipulation inherently occurs in a continuous space: the configuration space of a robot and the pose of objects in the environment are both continuous. If the state space S is continuous, then the belief space Δ is an infinite dimensional probability distribution. Representing a policy $\pi : \Delta \rightarrow A$ or a value function $V : \Delta \rightarrow \mathbb{R}$ over Δ is not straightforward unless $b(s_t)$ has a parametric form.

Our completed work avoided this problem by discretizing the



(a) Initial Belief State, $t = 0$ (b) $t = 1$ (c) $t = 7$ (d) $t = 15$

(e) Initial Belief State, $t = 0$ (f) $t = 1$ (g) $t = 2$ (h) $t = \infty$

Figure 5.1: Examples of discretization (a)–(d) introducing uncertainty and (e)–(h) causing overconfidence in belief. See the text for a detailed description.

We use $\phi^{-1}(\hat{s}) = \{s \in S : \phi(s) = \hat{s}\}$ to denote the pre-image of ϕ on S.

state space and solving a discrete POMDP (section 4.2). We use a discretization function $\phi : S \to \hat{S}$ induces a partition of the continuous state space S into discrete states $\hat{S}$. We compute discrete transition, observation, and reward functions by marginalizing the continuous functions over a probability distribution over the set of states $\phi^{-1}(\hat{s})$ contained in the current discrete state $\hat{s} \in \hat{S}$. We use the *principle of maximum entropy* to assume a uniform distribution over $\phi^{-1}(\hat{s})$ [Jaynes, 1957].

Unfortunately, contact produces belief states that are discontinuous (challenge 1). These belief states are poorly approximated by a distribution over $\hat{S}$ that is piecewise-constant over discrete states; i.e. maximum entropy is a poor assumption. As a result, planning in the discrete state space can produce a policy that is optimal over S_{discrete}, but *performs poorly in the underlying state space* S. We illustrate this in section 5.1.1, and propose possible solutions in sections 5.1.3 to 5.1.5.

5.1.1 *Discretization Artifacts*

Figure 5.1 shows two examples of discretization artifacts in a simple problem domain. The robot equipped with contact sensors on its fingertips and can move left, right, or forward in increments of 3 mm. The object is a point, drawn as a small black circle. The discrete state space S_{discrete} is a 1 cm grid and the discrete belief state is drawn by shading each cell with darkness proportional to its probability.

In fig. 5.1 (a)–(d), the robot moves right to perform an information-gathering action, then moves left to center the object in the hand. After (b) the first timestep, the discrete transition model $\hat{T}$, there is $2/3$ probability that the object remains in its initial state and $1/3$ probability that it transitions to the adjacent state. This uncertainty is eliminated when (c) the object contacts the sensor. Unfortunately, by (d) the time the information-gathering action is complete, enough uncertainty has accumulated to render it useless. This demonstrates that discretization can *introduce additional uncertainty* that is not present in

the underlying continuous model.

Discretization can also cause the robot to *become overly confident* about its state. In fig. 5.1 (e)–(h), the robot's contact sensor cannot differentiate between two discrete states. We assign $2/3$ and $1/3$ probability, respectively, to the lower and upper states that overlap with the sensor. In this situation, the discrete belief dynamics imply that the robot can localize the object in the lower state by (a) moving left, (b) moving right, and (c)–(d) repeating. In reality, the robot has not localized the object. This shows that discretization can *lead to overconfidence* compared to the underlying continuous model.

These discretization artifacts are more than a theoretical construction; they occur in practice when using the discretized belief dynamics to plan a policy. We partially addressed these issues in our completed work (section 4.2) by: (1) considering purely translational actions, (2) restricting actions to a length that is an integral multiple of the discretization resolution, and (3) explicitly discretizing the contact manifold. However, these assumptions are quite restrictive.

5.1.2 *Motivation for Discretization*

Our goal is to find a policy $\pi : \Delta \to A$ that maps a belief state to an action. We must choose a representation of π that allows us to: (1) optimize π and (2) evaluate $\pi[b(s_t)]$ efficiently.

In our completed work (section 4.2), we showed that contact decouples the pre- and post-contact policies (insight 2). This structure is critical to cope with the large observation branching factor $|O|$. We wish to choose a representation of π that can exploit this intuition; i.e plan efficiently by re-using the same sub-policies for multiple belief states without duplicating computation.

Additionally, once we have optimized π, we must evaluate $\pi[b(s_t)]$ online after receiving each observation. We wish to choose a representation of π that we can evaluate without requiring a large number of transition model evaluations. Querying the transition model is computationally expensive because it entails running a physics simulator (challenge 3).

If S was discrete, then we could represent a belief state $b \in \mathbb{R}^{|S_{\text{discrete}}|}$ as a vector. The value function of the optimal policy is piecewise-linear convex and can be written as

$$V^{\pi}[b] = \max_{\alpha_i \in \Gamma} \langle \alpha_i, b \rangle, \tag{5.1}$$

where each $\alpha_i \in \mathbb{R}^{|S_{\text{discrete}}|}$ is called an *α-vector* and $\langle \cdot, \cdot \rangle$ denotes the inner product [Smallwood and Sondik, 1973]. Figure 5.2 shows a graphical depiction of the relationship between α-vectors and V^{π}.

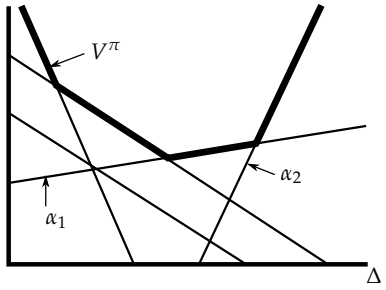


Figure 5.2: The optimal value function for a discrete POMDP is piecewise-linear convex and can be written as the maximum over a set of α-vectors. The value function of the policy induced by this value function is given by the bold sections of α-vectors.

We can efficiently point-based value iteration [Pineau et al., 2003] on sets of α-vectors to approximate π^*. Evaluating π^* is equivalent to executing the action associated with the α-vector that dominates V^{π^*} in the current belief state. This representation satisfies both of our requirements.

When S is continuous, eq. (5.1) still holds by replacing α-vectors with α-functions, b with a probability density function, and the maximum with a supremum [Porta et al., 2006]. We must choose parameterizations of b and α_i that allow us to perform a Bayes update and evaluate the inner product $\langle \alpha_i, b \rangle$. Unfortunately, the discontinuous nature of contact means that there is no straightforward choice of these parameterizations (challenge 1)

We propose three possible solutions. First, we can modify the point-based solver to allow us to discretize the state space at a much finer resolution than was possible in our completed work (section 5.1.3). Second, we can refine our discretization of the state space during planning to correct discretization artifacts (section 5.1.4). Finally, we can plan directly in the continuous space (section 5.1.5).

5.1.3 High Resolution Discretization

While the artifacts described in section 5.1.1 are fundamental to discretization, increasing the resolution of S_{discrete} will reduce the error introduced by discretizing S. Unfortunately, many implementations of discrete POMDP solvers assume that explicit models of $\hat{T}$ and $\hat{\Omega}$ are available. As we saw in our our completed work (section 4.2), it is only tractable to generate an these models when discretizing S at a relatively coarse resolution.

RQ 1. *How can we plan using a high resolution discrete POMDP solver model without generating the full discrete model in advance?*

We propose to modify an existing POMDP solver; e.g. SARSOP [Kurniawati et al., 2008], to lazily generate the discrete POMDP model when necessary for planning. For example, we can defer computation of the discrete transition probabilities for action a in state $\hat{s}$ until the solver evaluates $\hat{T}(s, a, \cdot)$ for the first time.

Our key insight is that a near-optimal policy will visit a relatively small subset of $\hat{S}$. For example, we do not expect the policy to push the object with the back of the hand. This is similar to the intuition behind the trajectory rollout representation of the contact manifold to focus samples on regions of the state space that we expect to encounter during execution of the optimal policy (section 4.1.2.2). We will use these results as a baseline of comparison for more sophisticated techniques.

5.1.4 *Adaptive Discretization*

Lazily building the discrete POMDP model allows us to discretize the state space at a higher resolution than is possible with an explicit model. However, uniformly discretizing the state space is still wasteful. High resolution discretization is only necessary when the policy leads to a continuous belief state $b(s_t)$ that differs significantly from the maximum entropy assumption.

Ideally, we would only finely discretize the regions of S that are necessary to represent π^*; i.e. the support of $\mathcal{R}^*[b_0]$. Finding the optimally reachable belief space $\mathcal{R}^*[b_0]$ is as hard as finding π^* [Kurniawati et al., 2008]. However, it is possible to approximate the set of states reached by π^* using a heuristic [Kurniawati et al., 2008] or upper and lower bounds [Paquet et al., 2005]. We propose to *adaptively discretize* S by interleaving planning with discretization.

RQ 2. *How can we refine our discretization of the state space over time to eliminate discretization artifacts?*

For example, we could begin by computing an optimal policy $\hat{\pi}_1^*$ using a coarse discretization of the state space $\hat{S}_1$. Next, we perform rollouts of $\hat{\pi}_1^*$ in the continuous state space and create a new discrete state space $\hat{S}_2$. The discretization has higher resolution in regions of S where the discrete value function $\hat{V}^{\hat{\pi}_1^*}$ deviates from the value function $V^{\hat{\pi}_1^*}$ estimated via rollouts. We can repeat this process to iteratively construct $\hat{S}_i$ from $\hat{S}_{i-1}$.

The key challenge to adaptive discretization is that an early policy $\hat{\pi}_i^*$ may not visit a region of S that must be finely discretized to find an optimal policy. We plan to build on existing literature on adaptive discretization for MDP planning [Sun and Reif, 2003].

5.1.5 *Policy Graph Representation*

Alternatively, we can avoid discretizing and directly compute the optimal policy π^* in the continuous space S. We propose to do so by representing π using a *policy graph* and using Monte Carlo value iteration to construct a policy offline [Bai et al., 2011].

A policy graph G is a directed acyclic graph where nodes are labeled with actions and edges are labeled with observations [Kaelbling et al., 1998]. The robot executes G by starting in some node v and executing the corresponding action $a_v \in A$. The robot receives observation $o \in O$ and follows the edge (v, v') labeled with o. This process repeats recursively from v' until the G is exhausted. At this point, the robot switches to executing a default policy π_{default}. An example policy graph is shown in fig. 5.3.

Each node v in a policy graph encodes one α-function. We can

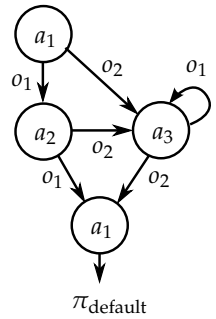


Figure 5.3: An example policy graph. Nodes are labeled with actions and edges are labeled with observations.

estimate the value function V^G for any belief $b \in \Delta$ by performing Monte Carlo rollouts of the policy encoded by G and recording the average reward. This makes it possible to directly perform value iteration on policy graphs [Bai et al., 2011], where each backup adds one node to G corresponding to the action with maximum value (fig. 5.4). Edges are inserted to connect the new node to successors nodes that maximize value. See Bai et al. [2011] for a more detailed description of Monte Carlo value iteration.

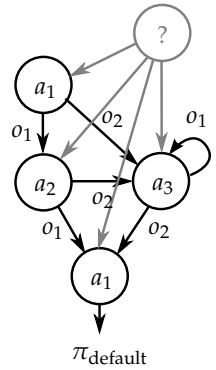


Figure 5.4: Performing a Bellman backup on a policy graph inserts one new node and up to $|O|$ new edges (both drawn in gray). The node and edge labels are select to maximize the value of the policy.

The policy graph representation is attractive for manipulation because: (1) it operates directly in the continuous state space S and (2) a policy graph can efficiently re-use sub-policies for multiple belief states. Unfortunately, manipulating a policy graph requires performing a large number of Monte Carlo rollouts. These rollouts are computationally expensive (challenge 3) and, since optimal kineodynamic motion planning is challenging, we do not have a good heuristic to guide exploration.

RQ 3. *How can we use the structure of manipulation to efficiently perform Monte Carlo value iteration?*

Just as in the discrete case, efficient planning depends on using upper and lower bounds on value to guide exploration of belief space. We can compute an upper bound $\bar{V}^\pi > V^\pi$ by solving an MDP that ignores uncertainty in state [Littman et al., 1995]. In our completed work, we further relaxed the problem by solving a determinstic problem that ignores uncertainty in transitions [Koval et al., 2015c]. We propose to approximate the value of this heuristic with a discrete lattice search. Since S is unbounded, we will lazily compute the heuristic only when queried by the planner.

5.1.6 *Evaluation*

The three proposed algorithms address the discretization artifacts we observed (section 5.1.1) when discretizing the state space in our completed work. As such, we will compare the proposed algorithms against a policy found using SARSOP [Kurniawati et al., 2008] using the discrete state space formulation from section 4.2. We expect the proposed algorithms to: (1) require less careful tuning of action durations, (2) allow for rotation of the hand, and (3) be capable of manipulating objects over a larger region.

5.2 *Manipulation with Kinematic Constraints*

Manipulation occurs in the joint configuration space of the robot Q and the object it is manipulating X. It is challenging to find a pol-

icy over S that generalizes between problem instances because Q is continuous and high-dimensional.

In our completed work on filtering [Koval et al., 2015b] and planning [Koval et al., 2015c], we bypassed this problem by considering the space of hand-relative object poses. Our key insight is that the transition and observation models depend only on the pose of the object relative to the hand S_{rel}, not the full state space $S = Q \times X$ (insight 3). If the goal G can also be expressed in S_{rel}, then we can plan in the lower dimensional space S_{rel} instead of the full space S.

Unfortunately, robots are often subject to *kinematic constraints* that cannot be represented in S_{rel}. For example, manipulators are frequently subject to *joint limits*. The links comprising the manipulator may also collide with the robot (*self collision*) or the environment (*environment collision*). These constraints can be encoded as a configuration-space obstacle S_c that depend on the full state S and cannot, in general, be represented in S_{rel}.

Planning in the full space S is challenging. However, as our experiments show, ignoring kinematic constraints while planning can generate a policies that perform poorly on the real robot (fig. 5.5). In this section, we propose to exploit the lower-dimensional structure for efficiency while considering kinematic constraints during planning.

Figure 5.5: HERB's elbow collides with the environment while localizing an object using a policy that ignores kinematic constraints. This photo is from a real-robot experiment presented in section 4.2.5.

5.2.1 *Lower-Dimensional Structure of the Problem*

Consider a POMDP $(S, A, O, T, \Omega, R, \gamma)$ with state space S, action space A, observation space O, transition model T, observation model Ω, reward function R, and discount factor γ. We define *projection functions* for state $\Lambda_S : S \to S_{\text{LD}}$, actions $\Lambda_A : A \to A_{\text{LD}}$, and observations $\Lambda_O : O \to O_{\text{LD}}$ that map from the original, high-dimensional POMDP to a low-dimensional POMDP with states S_{LD}, actions A_{LD}, and observations O_{LD}. Figure 5.6 shows a projection of a three-dimensional state space onto the plane.

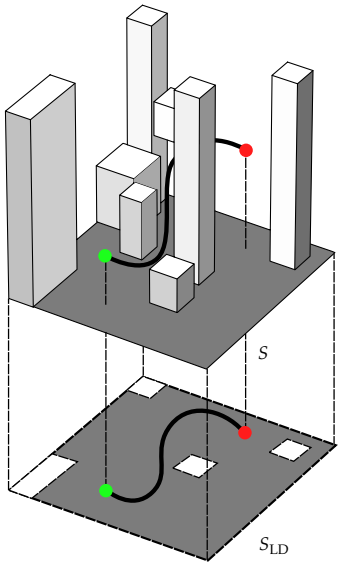


Figure 5.6: We project the POMDP into a lower-dimensional space. The low-dimensional state space S_{LD} does not include all constraints from the full-dimensional state space S.

We say that the lower-dimensional POMDP is an *projection* of the original POMDP if the low-dimensional model fully captures the behavior of the high-dimensional transition, observation, and reward functions. This property is satisfied if

$$\begin{aligned} T(s,a,s') &= T_{\text{LD}}(\Lambda_S(s), \Lambda_A(a), \Lambda_S(s')) && \forall s \in S, a \in A, s' \in S \\ \Omega(s,o) &= \Omega_{\text{LD}}(\Lambda_S(s), \Lambda_O(o)) && \forall s \in S, o \in O \\ R(s,a) &= R_{\text{LD}}(\Lambda_S(s), \Lambda_A(a)) && \forall s \in S, a \in A\,. \end{aligned}$$

If the property holds, we know that $V^{\pi}[b] = V^{\pi}_{\text{LD}}[\Lambda_\Delta[b]]$ where $\Lambda_\Delta[b]$ uses Λ_S to project a belief over S to a belief over S_{LD}. We can compute the optimal policy π^*_{LD} over the lower-dimensional problem

We use $\Lambda_A^{-1}(a_{\text{LD}}) = \{a \in A : \Lambda_A(a) = a_{\text{LD}}\}$ to denote the *pre-image* of Λ_A. An unique inverse does not exist because $\Lambda_A(\cdot)$ is not bijective.

by choosing the action that maximizes $V^{\pi^*}_{\text{LD}}$. Then, we project π^*_{LD} into the full-dimensional problem by selecting any action from the set $\Lambda_A^{-1}(\pi^*_{\text{LD}}[\Lambda_\Delta[b]])$.

5.2.2 *Projection into the Workspace*

In the case of manipulation, we define the high-dimensional POMDP in the joint configuration space $S = Q \times X$ and a lower-dimensional projection in the workspace (insight 3). In this section, we assume that Q is fully observed and has a deterministic transition model.

We define the lower-dimensional state $S_{\text{LD}} = SE(3)$ as the pose of the object relative to the end-effector (fig. 5.7). The corresponding projection is

$$\Lambda_S(s) = [\text{FK}(q)]^{-1}\, x$$
$$\Lambda_A(a) = J(q)a$$

where $q \in Q$ is the configuration of the robot, $x \in SE(3)$ is the pose of the object in the world frame, $\text{FK} : Q \to SE(3)$ is the forward kinematics of the manipulator, and J is the manipulator Jacobian.

We implicitly used this projection in our completed work [Koval et al., 2015c]. However, it is only valid under a restrictive set of circumstances. First, the object must only come in contact with the end-effector. Second, there must be no kinematic constraints on the manipulator. Finally, noise in T and Ω cannot depend on the configuration of the robot or the specific joint velocities being executed.

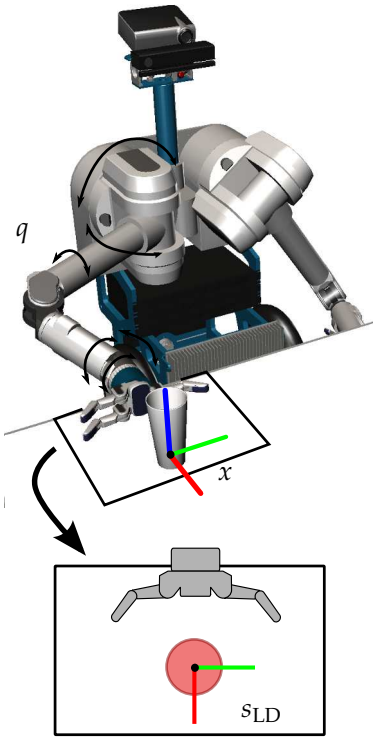


Figure 5.7: We plan in the space of hand-relative object poses, instead of the full joint configuration space $S = Q \times X$.

5.2.3 *Using the Projection as a Heuristic*

If the constraints listed above do not hold, e.g. if kinematic constraints are present, then we can only bound $V^\pi \leq V^\pi_{\text{LD}}$. In theory, introducing a kinematic constraint could completely change the optimal policy. However, our intuition is that the robot is often in relatively unconstrained parts of state space. As a result, we expect the optimal low-dimensional policy π^*_{LD} to closely resemble π^*.

RQ 4. *How can we use a pre-computed policy for a low-dimensional POMDP to efficiently find a good policy in higher-dimensional POMDP?*

Our key insight is to use the low-dimensional value function $V^{\pi^*}_{\text{LD}}$ to guide an online search for the optimal policy π^* in the full-dimensional problem. At a minimum, we can use π^*_{LD} as the *rollout policy* used to evaluate leaf nodes in our search tree [Silver and Veness, 2010, Kurniawati and Yadav, 2013, Somani et al., 2013]. If our solver is based on a Monte Carlo tree search, then we can additionally use $V^{\pi^*}_{\text{LD}}$ to bias the probability that we explore each sub-tree [Sil-

ver and Veness, 2010]. Variants of these two approaches have been successfully used in prior work [Gelly and Silver, 2007].

Additionally, we can exploit the low-dimensional structure of manipulation (insight 3) to minimize the number of times we evaluate the expensive transition and observation models (challenge 3). We only need to evaluate these models once per action in the pre-image $\Lambda_A^{-1}(\cdot)$. All other actions in this set will have the same impact on the low-dimensional state S_{LD} and observations O_{LD}.

5.2.4 *Evaluation*

Our completed work (section 4.2) and proposed work on planning in a continuous state space (section 5.1) ignore kinematic constraints while planning. We will compare the proposed algorithm against this as a baseline.

We expect the proposed algorithm to produce a policy with a higher success rate and/or achieve success more quickly when subject to kinematic constraints. Additionally, we may compare the proposed algorithm against the policy produced by an online search that does not use a low-dimensional policy to guide the search.

5.3 *Uncertainty in Proprioception*

Our completed (chapter 5) and our proposed work assumes that the goal $G \subseteq S$ is expressed in the hand frame. This allows us to ignore uncertainty in proprioception by defining state as the pose of the object relative to the hand. We cannot plan in this reduced space if: (1) the goal cannot be expressed in the world frame (fig. 5.8) or (2) uncertainty in proprioception causes the manipulator to contact the environment.

When these assumptions do not hold, we must plan in the joint configuration space $S = Q \times X$ of robot configurations and object poses. This is a challenging planning problem with a continuous, high-dimensional state space.

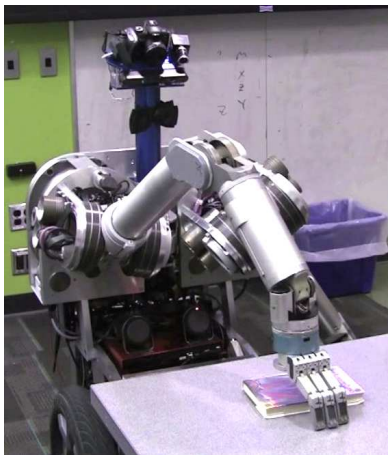


Figure 5.8: HERB pulling a book to the edge of the table as a form of pre-grasp manipulation. The goal region cannot be expressed in the hand frame.

5.3.1 *Information-Gathering Actions*

The robot naturally makes contact with the object while manipulating it. It also gains information about the pose of the object by forcing the object into the robot's contact sensors. As a result, we do not expect the optimal policy to stray far from contact when the goal is specified in the hand frame.

Unfortunately, this property does not hold if there is uncertainty in the configuration $q \in Q$ of the robot. The robot only gains information about q by taking explicit *information-gathering actions* that force it

into direct or indirect (through a movable object) contact with a static obstacle in the environment.

RQ 5. *How do we generate information-gathering actions that allow a closed-loop policy to reach the goal?*

If G is specified in the world frame, then we must localize the object in the world frame well enough to identify that it has reached G with high probability. We *do not* directly care about uncertainty in the configuration of the robot. However, uncertainty in q may affect our ability to successfully push the object into G. When this is the case, we must also reduce uncertainty in q.

Existing algorithms are successful at generating sequences of move-until-touch actions that reduce uncertainty in object pose [Hebert et al., 2013, Javdani et al., 2013] and localizing an object well enough to achieve the goal [Hsiao, 2009, Javdani et al., 2014]. We propose to use a similar algorithm to generate explicit information-gathering actions that reduce uncertainty on q. However, unlike in these approaches, our goal is only to drive q into a region that has high value under the policy we found in section 5.2. We can formulate these information-gathering actions as temporally extended *macro actions* [Hauskrecht et al., 1998]. Macro actions can be efficiently represented in a policy graph [Lim et al., 2011] and a similar approach has been successfully applied to grasping [Hsiao, 2009].

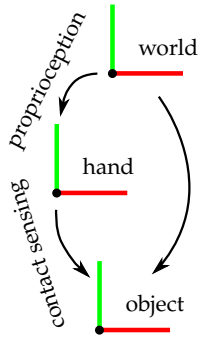


Figure 5.9: Successful manipulation depends on managing uncertainty in the world, end-effector, and object coordinate frames.

5.3.2 *Modelling Contact with the Environment*

Planning information-gathering actions for the manipulator is challenging because S is high-dimensional and continuous. As we showed in in our completed work (section 4.1), representing S using a set of weighted samples is difficult because the observation model Ω is peaked on the contact manifold during while receiving contact observations. The structure of the contact manifold becomes more complex when the robot may come into contact with static obstacles.

If the robot executes action $a \in A$ and comes in contact with the object, then the robot follows a and executes a position-controlled push. We assume that the robot immediately halts execution if it collides with S_c. We can implicitly detect that this occurred because robot's configuration will not match the expected outcome of a.

An observation $(o_q, o_c) \in O$ gives two important pieces of information (fig. 5.9). First, checking whether o_q matches the expected joint angles tests if the robot came into contact with a static obstacle. Second, o_c reveals whether a contact sensor came into contact with the environment. The contact sensor observation, unfortunately, does not explicitly reveal whether whether the sensor is touching the object or a static obstacle.

Note that S_{obs} is defined in the state space S, not the configuration space of the robot Q. It is possible for the robot to collide with S_{obs} by pushing against the movable object while it is resting against a static obstacle in the environment.

5.3.3 State Representation

Our key insight is that receiving an observation $o \in O$ that indicates contact constrains state in one of three ways (insight 1). First, contact between the end-effector the object constrains the pose of the object relative to the end-effector. Second, contact between the robot and a static obstacle in the environment constrains the configuration of the robot. Finally, pushing the object into a static obstacle enforces both of these constraints.

An observation $o \in O$ may indicate contact by: (1) including a contact observation from a contact sensor or (2) showing that the arm came to a stop while executing an action; i.e. hit a static obstacle.

RQ 6. *How do we efficiently represent a belief state over the joint configuration space of a robot and a movable object?*

We propose to factor the belief state as

$$b(s) = b(q, x) = b(q|x_{\text{rel}})b(x_{\text{rel}})$$

and represent $b(s)$ using a set of particles that are factored in a similar way. Each particle consists of (1) the pose of the object relative to the hand x_{rel} and (2) a set of robot configuration particles.

When the object is not in contact with the environment we assume that its motion is independent of the robot configuration q given the pose x_{rel} of the object relative to the hand. In this case, we propagate each object pose particle through the transition model only once, regardless of the number of robot configuration particles associated with it. Otherwise, we apply the transition model to each particle independently.

This is similar to our optimization in section 5.2. The key difference is that contact with the environment breaks our independence assumption.

To update our representation of $b(s)$, i.e. perform a Bayes update, we additionally assume that observing contact between the object and the end-effector refines *only our estimate of the pose of the object relative to the hand*; i.e. gives us no information about the robot configuration. This is an approximation: we could use these observations to refine our estimate of the full state S. However, this assumption allows us reduces the number of times we need to evaluate the transition model during planning.

Our representation is a simplified, approximate model of the true belief dynamics. This approximation allows us to use a large number of samples to track our belief over Q without increasing the number of (expensive) transition and observation model evaluations required during planning.

5.3.4 Evaluation Metrics

The proposed algorithm will enable us to plan for tasks where the goal is expressed in the world frame. We will evaluate the proposed algorithm on several of these tasks and compare against a baseline that does not consider uncertainty during planning.

6
Summary of Proposed Work

Timeline

Topic	Section	Questions	Deadline
Pose estimation for contact manipulation[1]	4.1	–	Mar. 2013 (IROS)
Discrete pre- and post-contact policy decomposition[2]	4.2	–	Jan. 2014 (RSS)
Thesis Proposal			May 2015
Discrete planning with adaptive discretization	5.1	RQ. 1 and 2	Sept. 2015 (ICRA)
Planning directly in continuous space	5.1	RQ. 1 and 3	Jan. 2016 (RSS)
Planning with kinematic constraints	5.2	RQ. 4	Mar. 2016 (WAFR)
Thesis Defense	–	–	Aug. 2016
Planning with proprioceptive error	5.3	RQ. 5 and 6	Sept. 2016 (ICRA)

Table 6.1: Proposed Timeline.
[1][Koval et al., 2013a,b, 2015b]
[2][Koval et al., 2014, 2015c]

Acknowledgements

This work was supported by the National National Science Foundation (award NFS-IIS-1218182), the Defense Advanced Research Projects Agency (award DARPA-BAA-10-28), the NASA Space Technologies Research Fellowship Program (award NNX13AL62H), and the Toyota Motor Corporation.

Bibliography

Dynamic Animation and Robotics Toolkit. http://dartsim.github.io, 2013.

A.-A. Agha-mohammadi, S. Chakravorty, and N.M. Amato. FIRM: Feedback controller-based information-state roadmap–a framework for motion planning under uncertainty. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011. DOI: 10.1109/IROS.2011.6095010.

M. Athans. The role and use of the stochastic linear-quadratic-Gaussian problem in control system design. *IEEE Transactions on Automatic Control*, 16(6):529–552, 1971. DOI: 10.1109/TAC.1971.1099818.

J.A. Bagnell, F. Cavalcanti, L. Cui, T. Galluzzo, M. Hebert, M. Kazemi, M. Klingensmith, J. Libby, T.Y. Liu, N. Pollard, et al. An integrated system for autonomous robotics manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

H. Bai, D. Hsu, W.S. Lee, and V.A. Ngo. Monte Carlo value iteration for continuous-state POMDPs. In *Workshop on the Algorithmic Foundations of Robotics*, 2011.

R. Bolles and R. Paul. The use of sensory feedback in a programmable assembly system. Technical Report STAN-CS-396, Computer Science Department, Stanford University, Stanford, CA, 1973.

B. Boots, A. Byravan, and D. Fox. Learning predictive models of a depth camera & manipulator from raw execution traces. In *IEEE International Conference on Robotics and Automation*, 2014.

M. Brokowski, M. Peshkin, and K. Goldberg. Curved fences for part alignment. In *IEEE International Conference on Robotics and Automation*, 1993. DOI: 10.1109/ROBOT.1993.292216.

E. Brunskill, L.P. Kaelbling, T. Lozano-Pèrez, and N. Roy. Continuous-state POMDPs with hybrid dynamics. In *International Symposium on Artificial Intelligence and Mathematics*, 2008.

E. Catto. Box2D. http://box2d.org, 2010.

E. Coumanns. Bullet physics library. http://www.bulletphysics.org, 2012.

H. Dang, J. Weisz, and P.K. Allen. Blind grasping: Stable robotic grasping using tactile feedback and hand kinematics. In *IEEE International Conference on Robotics and Automation*, pages 5917–5922, 2011. DOI: 10.1109/ICRA.2011.5979679.

R. Diankov and J. Kuffner. OpenRAVE: A planning architecture for autonomous robotics. Technical Report CMU-RI-TR-08-34, Carnegie Mellon University, Pittsburgh, PA, 2008.

M.A. Diftler, J.S. Mehling, M.E. Abdallah, N.A. Radford, L.B. Bridgwater, A.M. Sanders, R.S. Askew, D.M. Linn, J.D. Yamokoski, F.A. Permenter, et al. Robonaut 2–the first humanoid robot in space. In *IEEE International Conference on Robotics and Automation*, 2011.

M.R. Dogar. *Physics-Based Manipulation Planning in Cluttered Human Environments*. PhD thesis, Carnegie Mellon University, 2013.

M.R. Dogar and S.S. Srinivasa. Push-grasping with dexterous hands: Mechanics and a method. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010. DOI: 10.1109/IROS.2010.5652970.

M.R. Dogar and S.S. Srinivasa. A planning framework for non-prehensile manipulation under clutter and uncertainty. *Autonomous Robots*, 33(3):217–236, 2012. DOI: 10.1007/s10514-012-9306-z.

D.J. Duff. *Visual motion estimation and tracking of rigid bodies by physical simulation*. PhD thesis, University of Birmingham, 2011.

D.J. Duff, J. Wyatt, and R. Stolkin. Motion estimation using physical simulation. In *IEEE International Conference on Robotics and Automation*, 2010. DOI: 10.1109/ROBOT.2010.5509590.

M.A. Erdmann and M.T. Mason. An exploration of sensorless manipulation. *IEEE Journal of Robotics and Automation*, 1988. DOI: 10.1109/56.800.

K. Gadeyne, T. Lefebvre, and H. Bruyninckx. Bayesian hybrid model-state estimation applied to simultaneous contact formation recognition and geometrical parameter estimation. *International Journal of Robotics Research*, 24(8):615–630, 2005. DOI: 10.1177/0278364905056196.

J.D. Gammell, S.S. Srinivasa, and T.D. Barfoot. BIT*: Batch informed trees for optimal sampling-based planning via dynamic programming on implicit random geometric graphs. In *IEEE International Conference on Robotics and Automation*, 2015.

S. Gelly and D. Silver. Combining online and offline knowledge in UCT. In *International Conference on Machine Learning*, 2007. DOI: 10.1145/1273496.1273531.

N.J. Gordon, Salmond D.J., and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings F*, 1993.

P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 1968. DOI: 10.1109/TSSC.1968.300136.

M. Hauskrecht, N. Meuleau, L.P. Kaelbling, T. Dean, and C. Boutilier. Hierarchical solution of Markov decision processes using macro-actions. In *Conference on Uncertainty in Artificial Intelligence*, pages 220–âĂŞ229, 1998.

P. Hebert, T. Howard, N. Hudson, J. Ma, and J.W. Burdick. The next best touch for model-based localization. In *IEEE International Conference on Robotics and Automation*, 2013. DOI: 10.1109/ICRA.2013.6630562.

M. Horowitz and J. Burdick. Interactive non-prehensile manipulation for grasping via POMDPs. In *IEEE International Conference on Robotics and Automation*, 2013. DOI: 10.1109/ICRA.2013.6631031.

K. Hsiao. *Relatively robust grasping*. PhD thesis, Massachusetts Institute of Technology, 2009.

K. Hsiao, L.P. Kaelbling, and T. Lozano-Pèrez. Grasping POMDPs. In *IEEE International Conference on Robotics and Automation*, 2007. DOI: 10.1109/ROBOT.2007.364201.

K. Hsiao, T. Lozano-Pérez, and L.P. Kaelbling. Robust belief-based execution of manipulation programs. In *Workshop on the Algorithmic Foundations of Robotics*, 2008.

Y.K. Hwang and N. Ahuja. Gross motion planning–a survey. *ACM Computing Surveys*, 1992. DOI: 10.1145/136035.136037.

N. Hyafil and F. Bacchus. Conformant probabilistic planning via CSPs. In *International Conference on Automated Planning and Scheduling*, 2003.

S. Javdani, M. Klingensmith, J.A. Bagnell, N.S. Pollard, and S.S. Srinivasa. Efficient touch based localization through submodularity. In *IEEE International Conference on Robotics and Automation*, 2013. DOI: 10.1109/ICRA.2013.6630818.

S. Javdani, Y. Chen, A. Karbasi, A. Krause, J.A. Bagnell, and S.S. Srinivasa. Near optimal Bayesian active learning for decision making. In *Artificial Intelligence and Statistics*, 2014.

E.T. Jaynes. Information theory and statistical mechanics. *The Physical Review*, 106(4):620–630, 1957.

S.J. Julier and J.K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *International Symposium on Aerospace/Defense Sensing, Simulation, and Controls*, 1997.

L.P. Kaelbling and T. Lozano-Pérez. Integrated task and motion planning in belief space. *International Journal of Robotics Research*, 2013. DOI: 10.1177/0278364913484072.

L.P. Kaelbling, M.L. Littman, and A.R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 1998. DOI: 10.1016/S0004-3702(98)00023-X.

R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 1960. DOI: 10.1115/1.3662552.

S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, 2011. DOI: 10.1177/0278364911406761.

L.E. Kavraki, P. Švestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):556–580, 1996. DOI: 10.1109/70.508439.

M. Klingensmith, T. Galluzzo, C. Dellin, M. Kazemi, J.A. Bagnell, and N. Pollard. Closed-loop servoing using real-time markerless arm tracking. In *IEEE International Conference on Robotics and AutomationHumanoids Workshop*, 2013.

M.C. Koval, M.R. Dogar, N.S. Pollard, and S.S. Srinivasa. Pose estimation for contact manipulation with manifold particle filters. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013a. DOI: 10.1109/IROS.2013.6697009.

M.C. Koval, N.S. Pollard, and S.S. Srinivasa. Manifold representations for state estimation in contact manipulation. In *International Symposium of Robotics Research*, 2013b.

M.C. Koval, N.S. Pollard, and S.S. Srinivasa. Pre- and post-contact policy decomposition for planar contact manipulation under uncertainty. In *Robotics: Science and Systems*, 2014.

M.C. Koval, J.E. King, N.S. Pollard, and S.S. Srinivasa. Robust trajectory selection for rearrangement planning as a multi-armed bandit problem. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015a. Under review.

M.C. Koval, N.S. Pollard, and S.S. Srinivasa. Pose estimation for planar contact manipulation with manifold particle filters. *International Journal of Robotics Research*, 2015b. In press.

M.C. Koval, N.S. Pollard, and S.S. Srinivasa. Pre- and post-contact policy decomposition for planar contact manipulation under uncertainty. *International Journal of Robotics Research*, 2015c. Under review.

J.J. Kuffner and S.M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, 2000. DOI: 10.1109/ROBOT.2000.844730.

H. Kurniawati and V. Yadav. An online POMDP solver for uncertainty planning in dynamic environment. In *International Symposium of Robotics Research*, 2013.

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*, 2008.

S.M. LaValle and S.A. Hutchinson. An objective-based framework for motion planning under sensing and control uncertainties. *International Journal of Robotics Research*, 1998. DOI: 10.1177/027836499801700104.

W.S. Lee, N. Rong, and D.J. Hsu. What makes some POMDP problems easy to approximate? In *Advances in Neural Information Processing Systems*, 2007.

Q. Li, C. Schürmann, R. Haschke, and H. Ritter. A control framework for tactile servoing. In *Robotics: Science and Systems*, 2013.

Z.W. Lim, D. Hsu, and L. Sun. Monte Carlo value iteration with macro-actions. In *Advances in Neural Information Processing Systems*, 2011.

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

T. Lozano-Pèrez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, 1983. DOI: 10.1109/TC.1983.1676196.

T. Lozano-Pèrez, M. Mason, and R.H. Taylor. Automatic synthesis of fine-motion strategies for robots. *International Journal of Robotics Research*, 3(1):3–24, 1984. DOI: 10.1177/027836498400300101.

K.M. Lynch, H. Maekawa, and K. Tanie. Manipulation and active sensing by pushing using tactile feedback. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1992. DOI: 10.1109/IROS.1992.587370.

M.T. Mason. Mechanics and planning of manipulator pushing operations. *International Journal of Robotics Research*, 5(3):53–71, 1986. DOI: 10.1177/027836498600500303.

W. Meeussen, J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter. Contact-state segmentation using particle filters for programming by human demonstration in compliant-motion tasks. *IEEE Transactions on Robotics*, 23(2):218–231, 2007. DOI: 10.1109/TRO.2007.892227.

E. Nikandrova, J. Laaksonen, and V. Kyrki. Towards informative sensor-based grasp planning. *Robotics and Autonomous Systems*, 62(3):340–354, 2014. DOI: 10.1016/j.robot.2013.09.009.

L. Odhner, L.P. Jentoft, M.R. Claffee, N.Corson, Y. Tenzer, R.R. Ma, M. Buehler, R. Kohout, R.D. Howe, and A.M. Dollar. A compliant, underactuated hand for robust manipulation. *CoRR*, 2013. DOI: 10.1177/0278364913514466.

S. Paquet, L. Tobin, and B. Chaib-draa. An online pomdp algorithm for complex multiagent environments. In *International Joint Conference on Autonomous Agents and Multi Agent Systems*, 2005.

P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal. Online movement adaptation based on previous sensor experiences. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011. DOI: 10.1109/IROS.2011.6095059.

J. Pearl. *Heuristics: intelligent search strategies for computer problem solving*. Addison-Wesley, 1984.

A. Petrovskaya and O. Khatib. Global localization of objects via touch. *IEEE Transactions on Robotics*, 27 (3):569–585, 2011. DOI: 10.1109/TRO.2011.2138450.

J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence*, 2003.

R. Platt, A.H. Fagg, and R.A. Grupen. Nullspace grasp control: theory and experiments. *IEEE Transactions on Robotics*, 26(2):282–295, 2010a. DOI: 10.1109/TRO.2010.2042754.

R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Pèrez. Belief space planning assuming maximum likelihood observations. In *Robotics: Science and Systems*, 2010b.

R. Platt, L. Kaelbling, T. Lozano-Pèrez, and R. Tedrake. Simultaneous localization and grasping as a belief space control problem. In *International Symposium of Robotics Research*, 2011.

R. Platt., L.P. Kaelbling, T. Lozano-Pèrez, and R. Tedrake. Non-Gaussian belief space planning: Correctness and complexity. In *IEEE International Conference on Robotics and Automation*, 2012. DOI: 10.1109/ICRA.2012.6225223.

I. Pohl. *Practical and theoretical considerations in heuristic search algorithms*. University of California, Santa Cruz, 1977.

J.M. Porta, N. Vlassis, M.T.J. Spaan, and P. Poupart. Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research*, 7:2329–2367, 2006.

M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837, 1956.

S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 2008. DOI: 10.1613/jair.2567.

K. Salisbury, W. Townsend, B. Eberman, and D. DiPietro. Preliminary design of a whole-arm manipulation system (WAMS). In *IEEE International Conference on Robotics and Automation*, 1988. DOI: 10.1109/ROBOT.1988.12057.

A. Schneider, J. Sturm, C. Stachniss, M. Reisert, H. Burkhardt, and W. Burgard. Object identification with tactile sensors using bag-of-features. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009. DOI: 10.1109/IROS.2009.5354648.

D. Silver and J. Veness. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems*, 2010.

B.W. Silverman. Using kernel density estimates to investigate multimodality. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 97–99, 1981.

S.N. Simunovic. *An Information Approach to Parts Mating*. PhD thesis, Massachusetts Institute of Technology, 1979.

R.D. Smallwood and E.J. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, 21(5):1071–1088, 1973. DOI: 10.1287/opre.21.5.1071.

D.E. Smith and D.S. Weld. Conformant graphplan. In *National Conference on Artificial Intelligence*, 1998.

R. Smith. Open Dynamics Engine. `http://www.ode.org/`, 2007.

A. Somani, N. Ye, D. Hsu, and W.S. Lee. DESPOT: Online POMDP planning with regularization. In *Advances in Neural Information Processing Systems*, 2013.

S.S. Srinivasa, D. Berenson, M. Cakmak, A. Collet, M.R. Dogar, A.D. Dragan, R.A. Knepper, T. Niemueller, K. Strabala, and M. Vande Weghe. HERB 2.0: Lessons learned from developing a mobile manipulator for the home. *Proceedings of the IEEE*, 100(8):1–19, 2012.

F. Stulp, E. Theodorou, J. Buchli, and S. Schaal. Learning to grasp under uncertainty. In *IEEE International Conference on Robotics and Automation*, pages 5703–5708, 2011. DOI: 10.1109/ICRA.2011.5979644.

Z. Sun and J.H. Reif. Adaptive and compact discretization for weighted region optimal path finding. In *Fundamentals of Computation Theory*, pages 258–270, 2003.

Y. Tenzer, L.P. Jentoft, and R.D. Howe. Inexpensive and easily customized tactile array sensors using MEMS barometers chips. In *IEEE Robotics and Automation Magazine*, 2014.

S. Thrun, D. Fox, and W. Burgard. Monte Carlo localization with mixture proposal distribution. In *National Conference on Artificial Intelligence*, 2000a.

S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. Technical Report CMU-CS-00-125, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, 2000b.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT Press, 2005. DOI: 10.1145/504729.504754.

W. Townsend. The BarrettHand grasper–programmably flexible part handling and assembly. *Industrial Robot: An International Journal*, 27(3):181–188, 2000. DOI: 10.1108/01439910010371597.

J. van den Berg, P. Abbeel, and K. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. In *Robotics: Science and Systems*, 2010.

J. van den Berg, P. Abbeel, and K. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *International Journal of Robotics Research*, 31, 2011. DOI: 10.1177/0278364911406562.

P.M. Will and D.D. Grossman. An experimental system for computer controlled mechanical assembly. *IEEE Transactions on Computers*, 100(9):879–888, 1975. DOI: 10.1109/T-C.1975.224333.

J. Xiao. Automatic determination of topological contacts in the presence of sensing uncertainties. In *IEEE International Conference on Robotics and Automation*, 1993. DOI: 10.1109/ROBOT.1993.291962.

D. Xu, G.E. Loeb, and J.A. Fishel. Tactile identification of objects using Bayesian exploration. In *IEEE International Conference on Robotics and Automation*, 2013. DOI: 10.1109/ICRA.2013.6631001.

H. Zhang and N.N. Chen. Control of contact via tactile sensing. *IEEE Transactions on Robotics and Automation*, 16(5):482–495, 2000. DOI: 10.1109/70.880799.

L. Zhang and J.C. Trinkle. The application of particle filtering to grasping acquisition with visual occlusion and tactile sensing. In *IEEE International Conference on Robotics and Automation*, 2012. DOI: 10.1109/ICRA.2012.6225125.

L. Zhang, S. Lyu, and J.C. Trinkle. A dynamic Bayesian approach to simultaneous estimation and filtering in grasp acquisition. In *IEEE International Conference on Robotics and Automation*, 2013. DOI: 10.1109/ICRA.2013.6630560.